FIȘIERUL SUFERĂ MODIFICĂRI ZILNIC ȘI ESTE LICENȚIAT ASTFEL:

CC BY-NC-SA: This license allows reusers to distribute, remix, adapt, and build upon the material in any medium or format for noncommercial purposes only, and only so long as attribution is given to the creator. If you remix, adapt, or build upon the material, you must license the modified material under identical terms.

CC BY-NC-SA includes the following elements:

BY – Credit must be given to the creator

NC – Only noncommercial uses of the work are permitted

SA – Adaptations must be shared under the same terms

Curs de Informatică atractivă

cu exemple în C/C++ la nivel de liceu

Instalarea bibliotecii Conio2 va fi descrisă în subcapitolul 1.4.

Instalarea bibliotecii grafice WinBGIm este descrisă în capitolul 4.

Autor: Bogdan Pătruț

Clasa a IX-a

Capitolul 1. Elemente de bază

1. Instrucțiunea de atribuire
2. Instrucțiunea de decizie
3. Instrucțiunea for
4. Instrucțiunea while
5. Instrucțiunea do-while
6. Funcții (partea I)
7. Șiruri de caractere de tip String
8. O bibliotecă cu funcții de intrare și ieșire foarte utile

Capitolul 2. Aplicații la vectori și structuri

1. Funcții (partea a II-a)
2. Pisicile
3. Elevii
4. Pachetul de cărți de joc

Capitolul 3. Aplicații la șiruri de caractere și fișiere text

1. Jocul Spânzurătoarea (fără grafică)
2. Efecte de deplasare pe ecran a propozițiilor
3. Transformarea numerelor din scrisul cu cifre în scrisul cu litere

4. Despărțirea cuvintelor în silabe
5. Elisa - terapeut bazat pe sabloane, ca la inceputurile Inteligentei Artificiale.

Capitolul 4. Biblioteca grafică și exemple simple
1. O bibliotecă grafică simplă
2. Câteva exemple prezentate video
3. Desenarea unui poligon și a unei stele regulate
4. Figuri recursive cu cercuri si patrate
5. Graficul unei funcții

Capitolul 5. Aplicații cu grafică
1. Jocul Spânzurătoarea cu grafică
2. Ceas digital si analogic
3. Segmente
4. Morphing

Suplimentar:
5. Raliu auto
6. Formarea imaginilor în lentilele convergente/divergente

Capitolul 6. Aplicații pentru tablouri bi- și tridimensionale (I)
1. Jocul pătratelor glisante
2. Solitarul
3. Împărțirea careului
4. Colțul negru
5. Tom și Jerry
6. Mario pe scări

Suplimentar:
6. Cinci în linie
7. Câinii și vulpea
8. Război în 8
9. Jump In

Clasa a X-a

Capitolul 7. Aplicații pentru tablouri bi- și tridimensionale (II)
1. Pirates Hide and Seek
2. Gobblet Gobblers
3. Smart Farmer (varianta simplă / cu FILL)

Capitolul 8. Aplicații liste, stive și cozi
1. Calculator în limba română, engleză, franceză
2. Graficul unei funcții

Capitolul 9. Paradigme de elaborare a algoritmilor

1. Greedy
2. Divide et impera
3. Programare dinamică
   Analizor sintactic pentru limba romana (Algoritmul CKY)
4. Backtracking
5. Branch and bound
   Perspico
   Bonol sau L-Game

Clasa a XI-a

Capitolul 10. Aplicații structuri arborescente

1. Reprezentara grafică a unui arbore binar de căutare
2. Evaluator matematic și reprezentarea grafică a arborelui asociat expresiei
3. Derivare formală
4. Editor de expresii
5. Arbori Quad și vizualizator de imagini

Capitolul 10. Aplicații cu fișiere binare

1. Vizualizator de imagini BMP etc.
2. My Commander

Capitolul 12. Aplicații cu grafuri

1. Bridg-It
2. Algoritmica grafurilor
3. Quoridor

Clasa a XII-a

Capitolul 13. Aplicații diverse

1. Enciclopedie multimedia
2. BibMat - Bibliotecă de lucru cu tablouri (vectori și matrice)
3. BibLis. Bibliotecă de operații cu liste
4. Electron - Vizualizator de scheme electronice
https://www.youtube.com/watch?v=m5fvrgFDdGQ
5. Generator de diagrame Nassi-Schneiderman
6. InterSchem - Vizualizator și interpretor de scheme logice
7. Coduri Huffman, Algoritmul LZW (Lempel-Ziv-Welch) - Vizualizator de fisiere tip arhiva (TAR, ZIP)
8. Editor de texte - Realizarea unui editor simplu de texte, de genul Notepad.
9. Editor/vizualizator 3D

Clasa a IX-a

Capitolul 1. Elemente de bază

1.1. Instrucțiunea de atribuire

Probleme rezolvate

Să se scrie un program care să calculeze numărul de picioare dintr-o curte, în care se află g găini, p pisici şi un om.

Includem header-ul cu operaţiile de intrare şi ieşire standard:

```
#include <stdio.h>
```

Programul este constituit dintr-o singură funcţie, funcţia main:

```
int main()
{
      int g, p;
```

S-au declarat variabilele de intrare, care apoi se citesc cu scanf:

```
      printf("Dati numarul de gaini: ");
      scanf("%d",&g);
```

%d inseamnă "număr intreg"

```
      printf("Dati numarul de pisici: "); scanf("%d",&p);
```

&p înseamnă că valoarea lui p este preluată în funcţia scanf. În continuare, se declară şi se afişează data de ieşire, numărul total de picioare. Fireşte, o găina are două picioare, o pisică patru, iar omul are şi el două:

```
      int np=2*g+4*p+2;
```

Mai sus avem atât o declarare, cât si un calcul al variabilei np

```
      printf("\nNumarul total de picioare: %d",np);
      return 0;
}
```

"\n" de la începutul apelului lui printf reprezintă trecerea la un nou rând.
Apoi se termină funcția main și programul este gata.

Scrieți varianta C++ a acestui program, în care folosiți cin și cout din iostream, în loc de funcțiile standard printf și scanf din stdio.h.

Probleme propuse

Scrieți programe în C/C++ pentru a rezolva următoarele probleme. Toate variabilele se citesc de la tastatura, in ordinea in care apar in textul problemei.

1) Emilia are X pere și cu 9 mai multe mere. Ovidiu are Y pere și de 5 ori mai multe mere. Câte fructe au împreună Emilia și Ovidiu?

2) Carmen are X piersici (număr par) și Y mere. Petrica are 7 pere și cu N mere mai multe decât Carmen. Știind că Carmen îi dă lui Petrica jumătate din piersici, iar el îi dă ei două pere, să se determine câte fructe din fiecare tip va avea fiecare copil la final.

3) Lorand are X kiwi si Y pere. Linda are 18 kiwi și Z pere. Linda și Ioana au 9 mango, respectiv 19 mango. Tudor are cu N mango mai multe decât Linda. Știind că Linda îi dă lui Lorand jumătate din kiwi, să se determine câte fructe vor avea fetele și câte băieții la final?

4) Linda are X caise (număr par) și Y pere. Daniel are 6 piersici și cu N pere mai multe decât Linda. Știind că Linda îi dă lui Daniel jumătate din caise, iar el îi dă ei două piersici, să se determine câte fructe din fiecare tip va avea fiecare copil la final.

5) Bogdan are X portocale și Y mere. Ioana și Maria au 27, respectiv 63 cirese. Daniel are cu N fructe mai multe decât are Maria și Bogdan la un loc. Ioana are 9 mere. Știind că Ioana îi dă lui Daniel partea întreagă a unei treimi din cirese, să se determine câte fructe va avea fiecare fată și fiecare baiat, în parte, la final?

6) Maria are X portocale (număr par) și Y mango. Ionel are 7 caise și cu N mango mai multe decât Maria. Știind că Maria îi dă lui Ionel jumătate din portocale, iar el îi dă ei două caise, să se determine câte fructe din fiecare tip va avea fiecare copil la final.

1.2. Instrucțiuni de decizie

Probleme rezolvate

1) Să se verifice dacă un număr real x se află în intervalul [a,b).

```
#include <stdio.h>
int main()
{
```

Se declară cele trei variabile reale:

```
float a,b,x;
```

Se citesc capetele intervalului:

```
printf("Dati capetele intervalului:\n"); scanf("%f
%f",&a,&b);
```

Apoi se citeşte x:

```
printf("Dati numarul cu pricina:"); scanf("%f",&x);
```

Se verifică dacă x este în intervalul căutat: && înseamnă "şi"

```
if ((a<=x) && (x<b))
```

Dacă da, atunci se afişează textul: "*Numărul x se află...*" Valoarea lui x se afişează pe 6 poziţii, din care 3 zecimale (o poziţie o ocupă şi punctul zecimal)

```
else
    printf("Numărul %6.3f se află...",x);
```

... numărul nu se află în intervalul [a,b)...

```
    printf("Numărul %6.3f nu se află...",x);
    return 0;
}
```

2) Se dau trei numere reale a,b,c. Pot reprezenta ele lungimile laturilor unui triunghi?

Fireşte a, b, c pot fi lungimile laturilor unui triunghi doar dacă ele sunt toate numere pozitive, iar suma oricăror două este mai mare decât al treilea număr.

☞ Veţi observa cu uşurinţă că programul nici nu pare scris in limbajul C, ci mai degrabă într-o combinaţie de limba româna şi C. Și totuşi, programul merge şi face ce trebuie să facă. Tot secretul reuşitei lui constă în definiţiile de mai jos, în care relaţia "&&" este inlocuită cu cuvântul "si", în loc de "if" apare "daca", iar pentru "else" folosim "altfel".

```
#define si &&
#define daca if
#define altfel else
#include <stdio.h>
int main()
{
      float a,b,c;
      printf("Dati lungimile laturilor:\n");
```

```
	scanf("%f %f %f",&a,&b,&c);
	// Mai jos se verifică condiţia de triunghi:
	daca ((a>=0) si (b>=0) si (c>=0) si (a<b+c) si (b<a+c) si (c<a+b))
		printf("%4.2f, %4.2f si %4.2f formeaza triunghi.", a,b,c);
	altfel
		printf("Nu formeaza triunghi.");
	return 0;
}
```

3) Să se determine maximul şi minimul a două numere întregi, folosind macrocomenzi.

Programul ar fi banal, dacă am ști sau am avea voie să folosim instrucțiunea "`if`" (ce va fi predată ulterior). Vom utiliza interesantul operator condiţional "`? :`". Astfel, vom putea defini două macrouri, cu numele `minim` şi `maxim`, după cum urmează:

```
#define max(x,y) x > y ? x : y
#define min(x,y) x < y ? x : y
#include <stdio.h>
int main()
{
   int x, y;
   printf("Ia sa vedem care-i mai tare !\
n");
   printf("Da x si y:"); scanf("%d
%d",&x,&y);
   printf("\nCel mai tare este:
%d",max(x,y));
   printf("\nIar cel mai slab :
%d",min(x,y));
   return 0;
}
```

Simplu, elegant şi frumos, nu-i aşa?
Nu vă faceţi griji dacă lucraţi cu numere reale în loc de numere întregi, macrourile funcţionează perfect în orice situaţie!

```
#define max(x,y) x > y ? x : y
#include <stdio.h>
int main()
{
	float x, y;
	printf("Ia sa vedem care-i mai tare !\n");
	printf("Da x si y:"); scanf("%f %f",&x,&y);
	printf("\nCel mai tare este: %7.3f",max(x,y));
	return 0;
}
```

E simplu de înţeles acum cum funcţionează operatorul ternar "`?:`": să considerăm că avem 3 expresii e_1, e_2 şi e_3, în care e_2 şi e_3 trebuie să fie, în

mare fie spus, cam de acelaşi tip. Se evaluează e_1. Dacă e_1 e o valoare nenulă (*adevărat*), atunci rezultatul este e_2, altfel e_3.

4) Să se calculeze puterea a n-a a lui 2, unde n este un întreg dat de la tastatură.

☞ Pentru că tot am vorbit în exemplul anterior despre unul din operatorii speciali, pe care îi are limbajul C, vom prezenta în exemplul de faţă operatorul de “shiftare” ("şiftare", “rotire”, “deplasare”) spre stânga “<<”.
Fie x un număr întreg. Prin x<<1 înţelegem deplasarea spre stânga a biţilor din reprezentarea pe biţi (în baza 2) a lui x, completându-se cu 0 în dreapta. Astfel, de exemplu, dacă x=6, adică 101, atunci, deplasarea spre stânga va duce la obţinerea numărului x=12, adică 1010. De fapt, o shiftare spre stânga cu o unitate reprezinta o înmulţire a lui x cu 2. O shiftare spre stânga cu p poziţii a lui x reprezintă o înmulţire a lui x cu 2 de p ori.
Tot la fel, o shiftare spre dreapta ar însemna o împărţire succesivă la 2 (în care nu se ţine cont de rest).
Folosind acest artificiu (al deplasărilor spre stânga), am calculat şi noi puterea lui 2, conform programului următor:

```
#include <stdio.h>
int main()
{
      int n, m=1;
      //... aici se declara atât variabila n, cât şi m, care se şi iniţializează la valoarea 1
      printf("Dati n: ");
      scanf("%d",&n);
      printf("2^%d = %d\n",n,m=(m<<n));
      return 0;
}
```

Al doilea apel al funcţiei printf are un element interesant: se afişează valoarea expresiei “m=(m<<n)”.
Valoarea unei expresii de forma m=e, care evident este şi o instrucţiune de atribuire, este tocmai valoarea expresiei e. Astfel, când scriem m=(m<<n), înseamnă că m este shiftat spre stânga cu n poziţii, deci m, iniţial 1, devine egal cu puterea n a lui 2. Aceasta este şi valoarea expresiei m=(m<<n), care va fi afişată.

5) Să se determine aria unui triunghi în funcţie de lungimile laturilor sale.

Soluţia pe care o prezentăm în continuare va testa mai întâi dacă cele trei mărimi ar putea fi lungimile laturilor unui triunghi, lucru prezentat într-un exemplu anterior. Dacă da, vom folosi formula de calcul a ariei, când se cunosc lungimile a, b, c ale celor trei laturi, formulă cunoscută sub denumirea de *formula lui Heron*. Elementul interesant din acest program este folosirea

funcţiei `sqrt(x)`, care ne dă radăcina pătrată a unui număr real x. Folosirea acestei funcții presupune includerea fișierului <`math.h`>.

```
#include <stdio.h>
#include <math.h>

int main()
{
	float a,b,c,p,aria;
	printf("Dati lungimile laturilor:\n");
	scanf("%f %f %f",&a,&b,&c);
	if (a>0 && b>0 && c>0 && a<b+c && b<c+a && c<a+b) {
		p=(a+b+c)/2; aria=sqrt(p*(p-a)*(p-b)*(p-c));
		printf("Aria este: %7.3f.",aria);
	}
	else printf("Cele trei numere nu pot fi lungimile laturilor unui
triunghi.");
	return 0;
}
```

Probleme propuse

Scrieți programe în C/C++ pentru a rezolva următoarele probleme. Toate variabilele se citesc de la tastatura, in ordinea in care apar in textul problemei.

1) Cristina și Ioana sunt surori. Ele au împreuna N ani. Știind că Cristina are 12 ani, să se determine care dintre surori este mare și cu câți ani.

2) O firmă de transport aduce săptămânal, la un supermarket, A kilograme de fasole și B kilograme de ardei. Ana și Viorel au un mic magazin unde vând legume. Ei au nevoie zilnic de la supermarket de 6 kg de fasole și de 12 kg de ardei. Să se determine dacă supermarketul are suficiente legume săptămânal, pentru Ana și Viorel, știind că jumătate din fasole și 1/3 din ardei se vând altor clienți.

3) Un kilogram de cartofi costă P lei, iar un kilogram de fasole costă cu 3 lei mai mult.
Emilia dispune de S1 lei, Bogdan de S2 lei, iar Cristi de S3 lei. Emilia vrea să cumpere L1 (număr par) kilograme de cartofi, Bogdan vrea să cumpere L2 kilograme de fasole.
Cristi vrea să cumpere jumătate din cantitatea de cartofi cumpărată de Emilia și încă 4 kilograme. Să se afișeze, în ordine alfabetică, numele persoanelor care pot să-și cumpere legumele ce și le doresc.

1.3. Instrucțiunea for

Probleme rezolvate

1) Se citeşte numărul natural n de la tastatură. Să se afişeze triunghiul de numere:

```
1
1 2
1 2 3
.........
1 2 3  n
```

Rezolvare:

```
#include <stdio.h>
int main()
{
      int n,i,j; scanf("%d",&n);
      for (i=1; i<=n; i++)
            {
                  for (j=1; j<=i; j++)
                        printf("%2d",j);
                  printf("\n");
            }
       return 0;
}
```

2) Se citeşte numărul natural n de la tastatură. Să se afişeze triunghiul de numere:

```
1
1 3
1 3 5
.........
1 3 5  2n-1
```

Rezolvare:

```
#include <stdio.h>

int main()
{
      int n,i,j; scanf("%d",&n);
      for (i=1; i<=n; i++)
            {
                  for (j=1; j<=i; j++)
                        printf("%2d",2*j-1);
                  printf("\n");
            }
       return 0;
}
```

3) Se citeşte numărul natural n de la tastatură. Să se afişeze triunghiul de numere:

```
1 2 3 ........ n
```

```
1 2 3 ... n-1
1 2 3
1 2
1
```

Rezolvare:

```
#include <stdio.h>
int main()
{
      int n,i,j; scanf("%d",&n);
      for (i=n; i>=1; i--)
            {
                  for (j=1; j<=i; j++)
                        printf("%2d",j);
                  printf("\n");
            }
       return 0;
}
```

4) Se citeşte numărul natural n de la tastatură. Să se afişeze triunghiul de numere:

```
         1
        1 2
       1  2 3
      1  2  3 4
     ..........
   1 2  ........ n
```

Rezolvare:

```
#include <stdio.h>
int main()
{
      int n,i,j; scanf("%d",&n);
      for (i=1; i<=n; i++)
            {
                  for (j=1; j<=n-i; j++) printf("
                        ");
                  for (j=1; j<=i; j++)
                        printf("%4d",j);
                  printf("\n");
            }
       return 0;
}
```

5) Se citeşte numărul natural n de la tastatură. Să se afişeze triunghiul de numere:

```
n n-1     3 2 1
n n-1     2 1
............
n n-1
n
```

Rezolvare:

```
#include <stdio.h>
int main()
{
      int n,i,j; scanf("%d",&n);
      for (i=n; i>=1; i--)
          {
          for (j=n; j>=n-i+1; j--)
                printf("%2d",j);
          printf("\n");
          }
      }
      return 0;
 }
```

6) Pentru un număr n natural citit de la tastatură să se calculeze suma pătratelor perfecte mai mici decât n.

Rezolvare:

```
#include <stdio.h>
#include <math.h>
int main()
{
      long int n, suma, a; printf("\nDati n:");
      scanf("%ld",&n);
      for (suma=0, a=1; a<=sqrt(n); a++)
            if suma = suma + a*a;
      printf("Suma este:%ld",suma);
      return 0;
}
```

7) Se citeşte numărul natural n de la tastatură. Să se afişeze triunghiul de numere:

```
n  n-1   3 2 1
n-1 n-2  2 1
n-2 n-3   1
..........
3 2 1
2 1
1
```

Rezolvare:

```
#include <stdio.h>
int main()
{
      int n,i,j; scanf("%d",&n);
      for (i=n; i>=1; i--)
            {
                  for (j=i; j>=1; j--)
                        printf("%2d",j);
                  printf("\n");
```

```
            }
       return 0;
}
```

8) Se citeşte numărul natural n de la tastatură. Să se calculeze, folosind toate instrucţiunile repetitive cunoscute, expresiile:

- a) S = 1 + 2 + 3 + ... + n;
- b) P = 1 × 2 × 3 × ... × n;
- c) S = $1^3 + 2^3 + 3^3 + ... + n^3$;
- d) P = $1^2 \times 2^2 \times ... \times n^2$;
- e) S = 1 - 2 + 3 - 4 + ... ± n;
- f) S = 1×2 + 2×3 + 3×4 + ... + n×(n+1);
- g) S = 1 + 1/2 + 3 + 1/4 + ... (n termeni);
- h) P = 1 × 3 × 5 × ... × (2n-1);
- i) S = $1 + 3^2 + 5 + 7^2$ + ... (n termeni);
- j) S = 1 - 4 + 7 -10 + ... (n termeni);
- k) S = 1^2 - 1/2 + 3^2 - 1/4 + ... (n termeni);
- l) P = 1 × 4 × 7 × 10 × ... (n factori).

Rezolvare:

```
#include <stdio.h>
int main()
{
      int i,n,semn;
      long int suma, produs; float
      suma_r; printf("\nDati n:");
      scanf("%d",&n); printf("\nRezultate:\n");
// punctul a
      for (suma=0, i=1; i<=n; i++) suma+=i;
      printf("a) %ld\n",suma);
// punctul b
      for (produs=1, i=1; i<=n; i++) produs+=i;
      printf("b) %ld\n",produs);
// punctul c
      for (suma=0, i=1; i<=n; i++) suma+=i*i*i;
      printf("c) %ld\n",suma);
// punctul d
      for (produs=1, i=1; i<=n; i++) produs+=i*i;
      printf("d) %ld\n",produs);
// punctul e
      for (suma=0, semn=1, i=1; i<=n; i++, semn=-semn) suma+=semn*i;
      printf("e) %ld\n",suma);
// punctul f
      for (suma=0, i=1; i<=n; i++) suma+=i*(i+1);
      printf("f) %ld\n",suma);
// punctul g
      for (suma_r=0, i=1; i<=n; i++)
            if (i%2 ==1) suma_r=suma_r+i;
             else suma_r=suma_r+(float)1/i;
      printf("g) %0.2f\n",suma_r);
```

```
    // punctul h
       for (produs=1, i=1; i<=n; i++)
             produs*=(2*i-1);
       printf("h) %ld",produs);
       return 0;
}
```

9) Se citeşte numărul natural n de la tastaură. Să se determine suma primelor n numere prime.

Rezolvare:

```
#include <stdio.h>
#include <math.h>

long int prim(long int m)
{
      for (long int i=2; i<=sqrt(m); i++) if (m
            %i==0) return 0;
      return 1;
}
int main()
{
      long int n, suma, a, i; scanf("%ld",&n);
      for (suma=2, a=3, i=1; i<n; a+=2)
                  if (prim(a))
                        { suma+=a; i++; }
      printf("%ld",suma);
      return 0;
}
```

10) Se citeşte numărul natural n de la tastatură. Să se determine al n-lea termen din şirul lui Fibonacci şi suma primilor n termeni din acest şir.

Șirul lui Fibonacci este definit astfel: primii doi termeni sunt 1, iar oricare alt termen se calculează ca fiind suma celor doi termeni imediat anteriori: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 etc.

Rezolvare:

```
#include <stdio.h>
int main()
{
      long int t1, t2, t3, i, n, suma; printf("\nDati
      n:"); scanf("%ld",&n); t1=0; t2=1; t3=1;
      if (n==1) suma=1; else suma=2; for (i=2;
      i<n; i++)
            {
              t1=t2; t2=t3; t3=t1+t2; suma+=t3;
            }
      printf("Termenul=%ld, suma=%ld", t3, suma);
```

```
	return 0;
}
```

11) Pentru un număr n natural citit de la tastatură să se determine suma divizorilor săi pozitivi.

Rezolvare:

```
#include <stdio.h>
int main()
{
	long int n, i, suma;
	scanf("%ld", &n);
	for (i=1, suma=0; i<=n; i++) if (n % i
		== 0)
			suma+=i;
	printf("%ld",suma);
	return 0;
}
```

12) Pentru un număr n natural citit de la tastatură să se determine suma divizorilor săi pozitivi, care sunt numere impare.

Rezolvare:

```
#include <stdio.h>
int main()
{
	long int n, i, suma;
	scanf("%ld", &n);
	for (i=1, suma=0; i<=n; i++)
		if ((n % i == 0) & (i % 2 == 1))
			suma+=i;
	printf("%ld",suma);
	return 0;
}
```

Probleme propuse

1) Să se afişeze piramida de numere:

```
1
3 1
5 3 1
7 5 3 1
.................
(2n-1).....7 5 3 1
```

2) Să se calculeze, folosind instrucţiunea for, suma S=2+4+6+...+(2n), unde n>=1.

3) Să se calculeze, folosind instrucţiunea for, suma S=1+1x2+1x2x3+...
+1x2x3x...n, unde n>=1.

1.4. Biblioteca conio2.h

Se poate descărca de aici:
https://sourceforge.net/projects/conio/files/latest/download

Funcții din biblioteca conio2.h

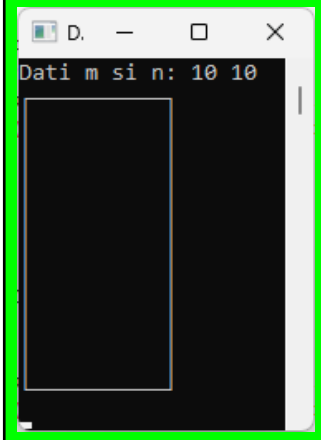
☞ aici

Instalarea bibliotecii conio2.h

☞ aici

Probleme rezolvate

1) Să se deseneze un dreptunghi de dimensiune m×n, folosind caracterele speciale pentru colţuri.



Vom folosi macroul putchar() pentru a afişa pe ecran. Colţurile au următoarele valori (in hexazecimal):

```
#include <stdio.h>
#include "conio2.h"
#define LEFT_TOP 0xDA
#define RIGHT_TOP 0xBF
#define HORIZ 0xC4
#define VERT 0xB3
#define LEFT_BOT 0xC0
#define RIGHT_BOT 0xD9
int main()
{
    int m, n;
    char i, j;
    clrscr();
    printf("Dati m si n: ");
    scanf("%d %d",&m,&n);
    /* marginea de sus */ putchar(LEFT_TOP);
    for (i=0; i<n; i++)
        putchar(HORIZ);
    putchar(RIGHT_TOP);
    putchar('\n');
    /* mijlocul */
    for (i=0; i<m; i++)
    {
        putchar(VERT);
        for (j=0; j<n; j++)
            putchar(' ');
        putchar(VERT);
        putchar('\n');
    }
    /* marginea de jos */ putchar(LEFT_BOT);
    for (i=0; i<n; i++)
        putchar(HORIZ);
    putchar(RIGHT_BOT);
```

```
        putchar('\n');
        getch();
        return 0;
    }
```

Veți putea observa ușor că și dacă dați aceeași valoare pentru m și n, atunci dreptunghiul nu va arăta ca un pătrat, deoarece dimensiunile pe verticală sunt mai mari decât cele pe orizontală.
În codul programului puteţi remarca utilizarea variabilelor `i`, `j` ca numere întregi, deşi ele au fost declarate ca fiind de tip `char`. Trebuie precizat că C-ul poate face astfel de conversii, iar un `char` ocupă doar un octet (iar un `int` doi), şi cum valorile pentru `i` şi `j` sunt mici, se poate folosi `char`.

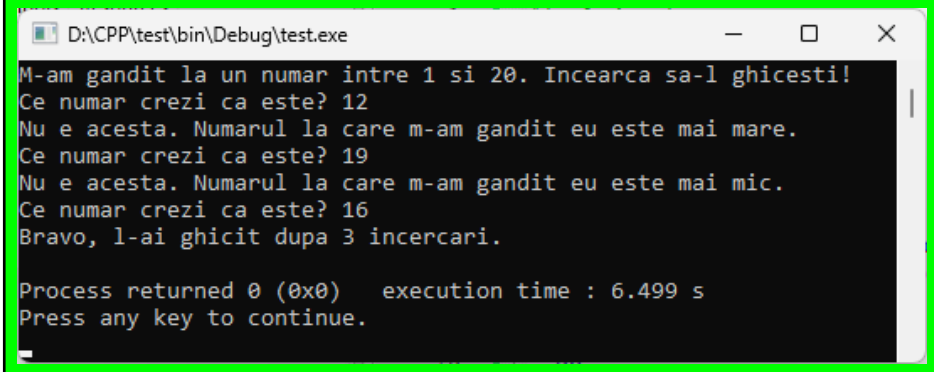
2) Să se producă în difuzor un sunet aleator, până se acţionează o tastă.

de testat dacă funcția Beep este și în conio2.

1.5. Instrucțiunile while și do-while

Probleme rezolvate

1) Jocul ghicitului unui număr

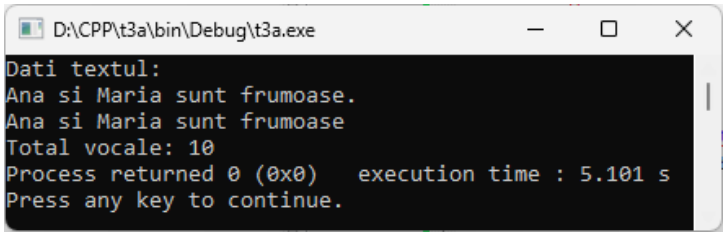



```
#include <iostream>
#include <time.h>

using namespace std;

int main()
{
  srand(time(NULL));
  cout<<"M-am gandit la un numar intre 1 si 20. Incearca sa-l ghicesti!"<<endl;
  int numarCalculator=1+rand()%20;
  int numarOm;
  int nrIncercari=0;
```

```
  do
  {
    cout<<"Ce numar crezi ca este? ";
    cin>>numarOm;
    nrIncercari++;
    if (numarOm==numarCalculator)
      cout<<"Bravo, l-ai ghicit dupa "<<nrIncercari<<" incercari."<<endl;
    else
    if (numarOm>numarCalculator)
      cout<<"Nu e acesta. Numarul la care m-am gandit eu este mai mic."<<endl;
    else
      cout<<"Nu e acesta. Numarul la care m-am gandit eu este mai mare."<<endl;
  } while (numarOm!=numarCalculator);
  return 0;
}
```

3) Se citesc litere pâna la intâlnirea caracterului ".". Să se contorizeze numărul de vocale citite.

☞ De data aceasta, vom folosi macroul predefinit `getchar()`, care returnează tasta apăsată, de fapt el citeşte din intrarea standard, `stdin`.




```
#include <stdio.h>
int main()
{
  char c; int nr_voc; nr_voc=0;
  printf("Dati textul:\n");
  while ((c = getchar()) != '.')
    {
          printf("%c", c);
          if ((c=='a') || (c=='e') || (c=='i') || (c=='o') || (c=='u'))
                nr_voc++;
    }
  printf("\nTotal vocale: %d",nr_voc);
  return 0;
}
```

Înlocuind '.' cu '\n', getchar() va citi până la apăsarea lui Enter.

4) Să se simuleze mişcarea unei bile de sus in jos, pe coloana 40 a ecranului. Afişarea se va repeta, de fiecare dată bila având o culoare aleasă la întâmplare.

Noutăţile aduse de acest exemplu sunt:

1. utilizarea fişierului <dos.h>, care conţine declaraţia funcţiei delay. Apelul delay(p) face o pauză (în execuţia programului) de p milisecunde. Astfel, delay(500) înseamnă o pauză de o jumătate de secundă.
2. utilizarea unor funcţii pentru obţinerea de numere aleatorii (de fapt generate după o anumită formulă, într-un şir de numere, pornind de la o anumită valoare iniţială):
   - apelul *srand(time(NULL))* schimbă valoarea iniţială a acestui şir de valori aleatoare
   - apelul *a=rand()%n* atribuie variabilei întregi *a* o valoare aleatoare întreagă, din intervalul [0,n-1]. Astfel, pentru a obţine una din cele 15 culori (exceptând negrul=0), am folosit textcolor(1+rand()%15)).
3. o altă noutate este folosirea lui puts pentru a afişa un şir de caractere;
4. răspunsul la întrebarea din finalul ciclului do-while este afişat pe ecran. El este o literă, preluată cu getche(). Dacă aceasta este ‘n’, programul se opreşte. Diferenţa între getche() şi getch() este că prima afişează caracterul apăsat (deci este cu “ecou”), pe când cea de a doua nu.

```
#include <stdio.h>
#include "conio2.h"
#include <time.h>

int main()
{
    srand(time(NULL));
    do {
        delay(500); clrscr(); textcolor(1+rand()%15);
        char y=1;
        while (y<=25) {
            gotoxy(40,y); putch(' ');
            gotoxy(40,++y); putch('o'); delay(50);
        }
        puts("\nMai doriti odata ?");
    } while (getche()!='n');
    return 0;
}
```

5) Să se simuleze pe ecran mişcarea unei bile pe diagonală, sub unghiul de 45 de grade, care să se comporte ca pe o masă de biliard, până la acţionarea unei taste.

Rezolvare:

Bila va fi caracterizată de patru variabile întregi:

- x şi y vor indica coordonatele bilei (1<=x<=80, 1<=y<=24);
- vx şi vy vor reprezenta direcţiile de deplasare ale bilei.

Astfel, la fiecare pas, bila se va deplasa din poziţia (x,y), în poziţia (x+vx, y+vy), unde vx şi vy pot lua valorile 1 sau -1. De pildă, o deplasare în direcţia stânga-jos este dată de vx=-1 şi vy=1.

Acest exemplu are următoarele elemente de noutate:

1. functia kbhit()- returnează 1 (adevărat) dacă fu apăsată o tastă (tocmai s-a apăsat o tastă), respectiv 0, în caz contrar;
2. se poate folosi funcția gotoxy pentru a poziționa locul de afișare (coloana și linia) și putch pentru a afișa un caracter pe ecran

```
#include "conio2.h"

int main()
{
    clrscr();
    char x=40, y=12, vx=1, vy=1;
    for (int i=1; i<=80; i++)
    {
        gotoxy(i,1);
        putch('*');
        gotoxy(i,25);
        putch('*');
    }
    for (int i=1; i<=24; i++)
    {
        gotoxy(1,i);
        putch('*');
        gotoxy(80,i);
        putch('*');
    }

    while (!kbhit())
    {
        gotoxy(x,y);
        putch(' ');
        if ((x==2) || (x==79)) vx=-vx;
        if ((y==2) || (y==24)) vy=-vy; x+=vx; y+=vy;
        gotoxy(x,y); putch('o'); gotoxy(1,1);
        delay(50);
    }

    return 0;
}
```

6) Să se mişte o literă pe ecranul text, folosind cele patru taste de cursor.

În exemplul anterior, am folosit afişarea cu putch() (care scrie doar pe

ecran, direct în memoria video). Poziţionarea cursorului se face cu funcţia gotoxy (declarată în conio.h). Vom considera un ecran cu 25 de linii şi 80 de coloane, numerotate de la 1 la 25, respectiv de la 1 la 80.

Observaţi şi utilizarea instrucţiunii de decizie multiplă switch. În cadrul ei am folosit instrucţiunea if pentru a evita ieşirile din ecran, precum şi instrucţiunea break, care determină continuarea execuţiei programului cu instrucţiunea de după switch.

Tasta acţionată se citeşte cu getch() (dar nu se şi afişează). Ea poate fi cursor sus (72), stânga (75), dreapta (77) sau jos (80), pentru deplasare, respectiv Escape (27), pentru terminarea programului.

```
#include <stdio.h>
#include "conio2.h"
#include <stdlib.h>

int main()
{
    clrscr();
    char tasta;
    char x=40,y=12;
    gotoxy(x,y);
    putch('X');
    gotoxy(1,1);
    do
    {
        tasta=getch();
        gotoxy(x,y);
        putch(' ');
        switch(tasta)
        {
        case 72:
        {
            if (y>1) y--;
            break;
        }
        case 80:
        {
            if (y<24) y++;
            break;
        }
        case 75:
        {
            if (x>1) x--;
            break;
        }
        case 77:
        {
            if (x<80) x++;
```

```
        }
        }
        gotoxy(x,y);
        putch('X');
        gotoxy(1,1);
    }
    while (tasta!=27);
    return 0;
}
```

7) Să se transforme un număr din baza 10 într-o bază p, 2<=p<10.

Vom împărţi succesiv pe x (numărul dat) la p, reţinând cifrele din baza p într-un număr xpi, care va fi apoi răsturnat (oglindit), pentru a obţine numărul xp, văzut ca reprezentarea în baza p a lui x. Pentru a nu avea probleme în cazul în care prima cifră a lui xpi ar fi 0, folosim o incrementare cu 1 a cifrelor la transformarea xxpi, respectiv o decrementare la transformarea xpixp.

```
#include <stdio.h>
int main()
{
  int x, xpi, xp, p;
  printf("Dati x in baza 10: "); scanf("%d",&x);
  printf("Dati noua baza p : "); scanf("%d",&p);
  for (xpi=0; x; xpi=10*xpi+x%p+1, x=x/p);
  for (xp=0; xpi; xp=10*xp+(xpi-1)%10, xpi=xpi/10);
  printf("Numarul in baza %d este: %d",p,xp);
  return 0;
}
```

De remarcat, în acest exemplu, eleganţa folosirii operatorului “,” în cadrul instrucţiunilor for. De pildă, primul for se interpretează astfel:

- se iniţializează xpi cu 0;
- cât timp x este diferit de 0 execută:
    - xpi devine 10*xpi+x%p+1;
    - x se împarte la p (şi se face conversia la int)

8) Să se calculeze suma S = 1-2+3-4+...n, pentru un n dat de la tastatură.

Vom folosi o variabilă semn care va lua pe rând valorile 1 şi -1.

```
#include "conio2.h"
#include <stdio.h>
void main()
{
  clrscr(); int n;
  printf("Dati numarul n: "); scanf("%d",&n);
  for (int i=1, suma=0, semn=1; i-(n+1); suma+=(semn*i), i++, semn=-semn);
  printf("Suma dorita este: %d",suma); getch();
}
```

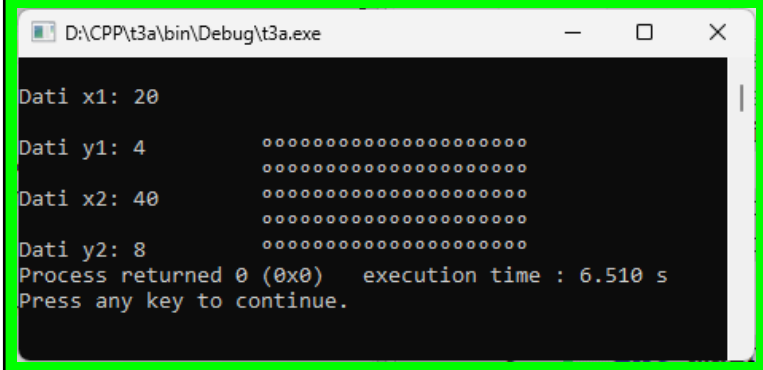
Fireşte, suma e foarte uşor de determinat matematic. Am vrut, însă, să punem în evidenţă următoarele aspecte:

- declararea şi iniţializarea variabilelor i, suma şi semn, în cadrul primei expresii din instrucţiunea for;
- folosirea condiţiei i-(n+1) pentru a verifica dacă i<=n sau nu;
- folosirea operatorului virgulă în cadrul expresiilor ce apar in “for”;
- inexistenţa unei alte instrucţiuni în cadrul lui “for”.

De remarcat puterea de care dă dovadă instrucţiunea for din limbajul C, în comparaţie, de pildă, cu cea din cazul limbajului Pascal.

10) Să se deseneze în ecranul text un dreptunghi, specificat prin coordonatele a două colţuri diagonal opuse.

Vom citi x1, y1 şi respectiv x2, y2, coordonatele acestor colţuri. Folosind operatorul ternar “?:” vom determina în x1, y1 minimele, iar în x2, y2 maximele (ceea ce va însemna că, în final, aceste coordonate vor fi ale colţurilor stânga-sus şi dreapta-jos). Apoi vom folosi două instrucţiuni for pentru a desena. Caracterul va avea codul hexazecimal B0 (adică 178) şi va fi afişat cu o funcţie PrintAt.




```
#include "conio2.h"
#include <stdio.h>
void PrintAt(char x, char y, char c)
{
    gotoxy(x,y); putch(c);
}

int main()
{
    int x1,y1,x2,y2,aux,x,y;
    char cc=0xB0; // numar hexazecimal = 178, in baza 10
    // clrscr();
    printf("\nDati x1: ");
    scanf("%d",&x1);
    printf("\nDati y1: ");
    scanf("%d",&y1);
    printf("\nDati x2: ");
    scanf("%d",&x2);
```

```
    printf("\nDati y2: ");
    scanf("%d",&y2);
    // Daca x1>x2, folosind variabila aux, se interschimba valorile lui x1
si x2
    x1>x2 ? aux=x1, x1=x2, x2=aux: 1;
    //... aici, in loc de 1 ar fi putut fi orice; Facem la fel pentru y1 si
y2:
    y1>y2 ? aux=y1, y1=y2, y2=aux: 1;
    // clrscr();
    for (x=x1; x<=x2; x++)
        for (y=y1; y<=y2; y++) PrintAt(x,y,cc);
//... se face conversie de la int la char
    return 0;
}
```

Probleme propuse

Aici vor fi probleme propuse

1.6. Funcții (partea I)

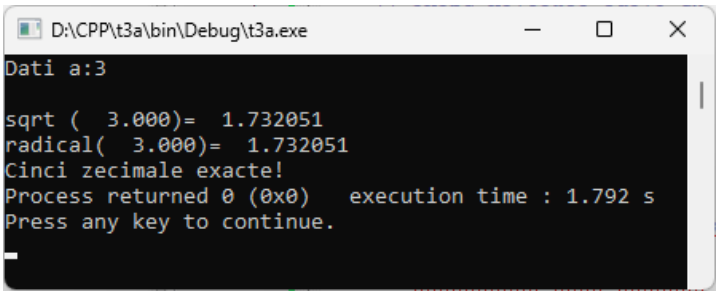
Probleme rezolvate

1) Să se determine rădăcina pătrată a unui numar real a, fără a utiliza funcţia sqrt.

Această problemă se poate soluţiona pornind de la *şirul recurent al lui Newton*:

$x_0 = 1, x_n = (x_{n-1} + a/x_{n-1})/2$, pentru n1

De fiecare dată vom avea nevoie doar de doi termeni succesivi ai acestui şir, fie ei xn şi xn1. Vom calcula pe xn plecând de la xn1, apoi vom actualiza pe xn1 cu valoarea lui xn, iar procedeul continuă repetitiv până când diferenţa dintre xn şi xn1 este nesemnificativă.




```
#include <stdio.h>
#include <math.h>

float radical(float a)
{
```

```
    const eps=0.00001; // o constanta,
    // avand valoarea egala cu precizia dorita in calculul radicalului
    float xn1=1, xn=(1+xn1)/2;
    do
    {
        xn1=xn;
        xn = (xn1+a/xn1)/2.0;
    }
    while (fabs(xn-xn1)>eps);
    // fabs(x) = |x|, unde x este numar real
    return xn;
    // rezultatul este ultimul termen calculat al sirului
}
int main()
{
    float a;
    printf("Dati a:");
    scanf("%f",&a);
    printf("\nsqrt (%7.3f)=%10.6f",a,sqrt(a));
    // se compara cele doua functii
    printf("\nradical(%7.3f)=%10.6f",a,radical(a));
    printf("\nCinci zecimale exacte!");
    return 0;
}
```

☞ Observaţi folosirea instrucţiunii repetitive do... while... , având semnificaţia: *execută ... atât timp cât ...* . Este vorba despre o instrucţiune repetitivă în care testul se face la sfârşitul ciclului.

2) Să se determine cel mai mare divizor comun şi cel mai mic multiplu comun pentru două numere naturale a şi b.

Pentru a determina cel mai mic multiplu comun al lui a şi b, trebuie ca mai întâi să determinăm c = cel mai mare divizor comun al lui a şi b. Dacă cel puţin unul din cele două numere (a sau b) este 0, vom considera celălalt număr ca fiind c. Dacă ambele numere sunt diferite de 0, atunci vom scădea numărul mai mic din numărul mai mare, până vom obţine egalitatea numerelor a şi b. În acest moment, c=a. În fine, metoda se numeste **algoritmul lui Euclid cu scaderi**, iar pentru a obţine cel mai mic multiplu comun înmulţim pe a şi b iniţiali şi împărţim la c.

```
  #include <stdio.h>
  #include "conio2.h"
  long cmmdc(long a, long b)
  {
    if (!a) return b;
    else if (!b) return a;
                else while (a-b) // adica daca a-b != 0, adica a !=b if
                (a>b) a-=b; else b-=a;
    return a;
```

```
}
long cmmmc(long a, long b)
{ long c=cmmdc(a,b);
  if (c) return a*b/c; else return 0;
}
```

Când ambele numere sunt 0, cmmmc nu există, dar returnăm 0.

```
int main()
{
  long a,b,c,d;
  printf("\nDati a, b: "); scanf("%ld %ld",&a,&b); c=cmmdc(a,b);
  d=cmmmc(a,b); // conversia rezultatului impartirii
  // la numarul intreg d se face automat printf("Cmmdc(%ld,%ld)=%ld\
  n",a,b,c); printf("Cmmmc(%ld,%ld)=%ld\n",a,b,d); return 0;
}
```

De subliniat faptul că o declaraţie de forma `long x` este echivalentă cu o declaraţie de forma `long int` (întreg lung). `"%ld"` permite afişarea de întregi lungi.

3) Să se scrie un program care să verifice dacă trei numere reale a, b şi c pot reprezenta lungimile laturilor unui triunghi. Dacă da, atunci să se precizeze de ce tip este triunghiul: echilateral, isoscel, dreptunghic sau oarecare.

De data aceasta, dacă tot am scris programul anterior, în care verificăm dacă a, b, c pot fi lungimile laturilor unui triunghi, vom rescrie funcţia main() din acel program, dându-i numele FormeazaTriunghi în acest nou program.

☞ Astfel, când în funcţia main() se va ajunge la instrucţiunea if FormeazaTriunghi(a,b,c)..., programul îşi va continua execuţia cu primul rând din funcţia sus numită. În locul lui x (primul parametru al funcţiei) va veni a, adică valoarea sa, în locul lui y va veni valoarea lui b, iar in locul lui z - valoarea lui c. Fireşte, parametrii (argumentele) funcţiei FormeazaTriunghi ar putea fi numite chiar şi a, b, c, (eventual în altă ordine), aceste variabile neavând nimic comun cu variabilele a,b,c din funcţia main() (în afară de nume).

☞ Se spune că apelul funcţiei se face **prin valoare**. Comunicarea main()-FormeazaTriunghi() este într-un singur sens, de la prima funcţie către a doua. Astfel, chiar dacă, să zicem, x s-ar modifica în interiorul celei de a doua funcţie, aceasta nu ar afecta în nici un fel valoarea variabilei a, adică a parametrului efectiv cu care main() apelează pe FormeazaTriunghi(). Puteţi verifica aceasta singuri sau urmăriţi exemplul următor. Funcţia FormeazaTriunghi returnează o valoare de adevăr, adică *adevărat* sau *fals*,

lucru care se traduce in limbajul C printr-o valoare nenulă, pentru *adevărat*, respectiv prin valoarea 0, pentru *fals*. Variabilele echil, isos şi drept au valorile 1 dacă triunghiul este respectiv echilateral, isoscel, dreptunghic. Fireşte, un triunghi echilateral va fi şi isoscel. E greu să se obţină triunghiuri dreptunghic iscoscele, deoarece e greu să se introducă (pentru scanf()) valori potrivite.

```
#include <stdio.h>

int FormeazaTriunghi(int x, int y, int z)
{
    return ((x>=0) && (y>=0) && (z>=0) && (x<y+z) && (y<x+z) && (z<x+y));
}

int main()
{
      float a,b,c;
      printf("Dati lungimile laturilor:\n"); scanf("%f %f
      %f",&a,&b,&c);
      if (FormeazaTriunghi(a,b,c))
      {
        int echil=((a==b) && (b==c));
        int isos=((a==b) || (b==c) || (c==a)); int
        drept=((a*a==(b*b+c*c)) ||
        (b*b==(a*a+c*c)) || (c*c==(a*a+b*b))); if (echil)
        printf("Triunghi echilateral.\n"); if (isos)
        printf("Triunghi isoscel.\n");
        if (drept) printf("Triunghi dreptunghic.\n"); if ((!
        echil) && (!isos) && (!drept))
              printf("Triunghi oarecare.\n");
      }
      else printf("Nu formeaza triunghi.\n");
      return 0;
}
```

4) Să se scrie un program care să returneze succesorul unui număr intreg dat.

Este vorba despre un program banal, însă vom încerca să punem în evidenţă un anumit aspect, anume o simulare a transmiterii prin valoare a parametrilor, în cazul funcţiilor.

```
#include <stdio.h>
int succesor(int x)
{
      int y=x+1;
      // ... evident, y este succesorul lui x.
      x++;
       // ... îl creştem pe x cu o unitate.
       // Oare se va transmite în exterior valoarea nouă?
      return y;
}
```

```
int main()
{
      int a;
      printf("Dati numarul a: "); scanf("%d",&a);
      printf("Deocamdata a = %d\n",a);
      int b=succesor(a);
      printf("Succesorul b = %d\n",b);
      printf("Iar acum a = %d\n (ala vechi!)\n",a);
      int c=succesor(3);
      printf("Succesorul lui 3 este = %d\n",c); printf("Iar 3
      tot 3 ramane, nu-i asa ?\n"); int d=succesor(a+3);
      printf("In fine, succesorul lui 3+%d este = %d\n",a,d);
      return 0;
}
```

Se va observa cu usurinţă, că valoarea modificată a lui x nu se transmite şi lui a. Motiv pentru care puteţi apela funcţia succesor şi cu un argument constant sau cu o expresie ca în exemplul anterior. Iată de ce e nevoie de un nou mecanism de transmitere a parametrilor, pentru a realiza comunicarea de la funcţia apelată către cea apelantă (de exemplu, de la succesor la main).

Un astfel de mecanism se numeşte **transmitere prin referinţă a parametrilor** şi aceasta se realizează în felul următor: în loc să punem ca argument formal în apel numele variabilei respective, vom pune simbolul "&" în faţa variabilei. Aceasta schimbă semnificaţia! Dacă x este o variabilă oarecare, prin &x vom înţelege adresa din memorie unde se află variabila x (indiferent pe unde şade ea acolo!). Astfel, trebuie ca şi funcţia care primeşte pe &x ca argument să ştie să lucreze cu o adresă a unei variabile de tipul lui x (practic este vorba tot despre un apel prin valoare, insa valorile sunt adrese de memorie sau **pointeri**).

Un exemplu îl constituie problema următoare.

5) Să se scrie un program care să returneze succesorul unui număr întreg dat, dar să încrementeze cu 1 valoarea numărului iniţial.

Vom încerca sa punem în evidenţă un anumit aspect, anume transmiterea prin referinţă a parametrilor, în cazul funcţiilor

```
#include <stdio.h>
int succesor(int *x)
{
      int y=(*x)+1;
      // ... evident, y este succesorul valorii lui x
      (*x)++;
      //... creştem valoarea lui x cu o unitate.
```

```
	return y;
}

void main()
{
	int a;
	printf("Dati numarul a: "); scanf("%d",&a); printf("Deocamdata a = %d\
n",a);
	int b=succesor(&a); printf("Succesorul b = %d\n",b);
	printf("Iar acum a = %d\n (ala vechi!)\n",a);
	// printf("Succesorul lui 3 = %d",succesor(&3);
}
```

Funcţia succesor ar fi putut arăta şi astfel:

```
int succesor(int *x)
{
	return ++(*x);
}
```

dar nu şi aşa:

```
int succesor(int *x)
{
	return (*x)++;
}
```

Fireşte, atât ++m, cât şi m++ reprezintă incrementări ale lui m cu o unitate, însă diferenţa e de fineţe: pe când în primul caz are loc întâi incrementarea lui m, apoi folosirea sa, în cel de al doilea caz are loc întâi utilizarea sa (a valorii neincrementate încă) şi apoi incrementarea. Atenţie deci, căci puteţi avea probleme! Se va observa că valoarea lui x se transmite în exterior variabilei a. Înainte de a trece mai departe, să analizăm înca puţin exemplul dat. Dacă până acum noi puteam apela funcţiile descrise nu doar cu variabile, ci şi cu constante sau chiar expresii (în fond cu valorile lor), de această dată, un apel de genul celui pus în comentariu în finalul programului anterior semnaleaza eroare. Nici apelul succesor(3) nu e mai bun! Dar, dacă &m înseamnă adresa variabile m, acest lucru ne obligă ca în antetul funcţiei care o foloseşte, să definim ceva de genul nume_functie(tip * mm), în care tip este tocmai tipul lui m, mm este argumentul funcţiei, iar steluta “*” semnalează faptul că mm nu este un număr întreg, ci un pointer la un număr întreg, adică un indicator către o zonă de memorie unde se află un număr întreg (practic, în momentul execuţiei programului, apelând funcţia cu valoarea adresei lui m (&m), mm va conţine tocmai această adresă (ceea ce se numeşte că mm este un “indicator” către m)

Oare funcţia scanf() nu lucrează la fel? Ba da, e şi normal, deoarece ea trebuie să trimită către exterior (adică funcţiei apelante) rezultatele citirilor, deci întotdeauna trebuie să avem grijă cum folosim funcţia scanf().

Fiindcă printf() nu modifică în nici un fel valorile afişate, e normal ca la printf() să nu avem nevoie de &. Decât, dacă dorim să obţinem adresa unei variabile, ca în exemplul următor:

```
#include <stdio.h>
void main()
{ int x=3; printf("Adresa lui x: %ld",&x); }
```

Dar un asemenea exemplu nu ne ajută cu nimic, de obicei, fiind pur didactic...

6) Să se scrie un program care să calculeze valoarea mediei aritmetice a două valori reale a şi b. Se vor folosi două funcţii, una care va trimite rezultatul prin numele funcţiei, iar alta printr-unul din parametri.

```
#include <stdio.h>
float media(float x, float y)
{
      return (x+y)/2;
}
void proc_media(float x, float y, float * m)
{
      *m = (x+y)/2;
}
int main()
{
      float a, b, m1, m2;
      printf("Dati cele doua numere:\n");
      scanf("%f %f",&a,&b);
      m1 = media(a,b);
proc_media(a,b,&m2);
      printf("Media returnata prin numele functiei: %7.3f\n", m1);
      printf("Media returnata ca si parametru : %7.3f\n", m2);
      printf("Nici o diferenta !\n"); return 0;
}
```

De fapt, dacă prima funcţie foloseşte return pentru a obţine valoarea medie (adică se comportă ca o funcţie “adevarată”), cea de a doua se comportă ca o procedură din limbajul Pascal, motiv pentru care nici nu returnează nimic (apare void ca tip al funcţiei), iar rezultatul se transmite prin intermediul parametrului de tip pointer m.

☞ Observaţi diferenţa dintre x şi y, pe de o parte, şi m pe de altă parte, în antetul funcţiei proc_media, cât şi modul în care este apelată ea!

```
#include <iostream>
using namespace std;

int sumaCifre(int numar)
```

```
{
  int suma=0, cifra;
  while (numar > 0)
    {
      cifra = numar % 10;
      suma += cifra;
      numar /= 10;
    }
  return suma;
}

int oglindit(int numar)
{
  int og=0, cifra;
  while (numar > 0)
  {
    cifra = numar % 10;
    og = og * 10 + cifra;
    numar /= 10;
  }
  return og;
}

int main()
{
  int numar;
  cin>>numar;
  cout<<oglindit(sumaCifre(numar));
  return 0;
}
```

7) Scrieți un program care să verifice dacă un număr este prim.

```
#include <iostream>

using namespace std;

bool estePrim(int numar)
{
  if (numar < 2)
    return false;
  else
    {
      int divizor;
      for (divizor = 2; divizor * divizor <= numar; divizor++)
        if (numar % divizor == 0)
          return false;
      return true;
    }

}

int main()
{
```

```
  int numar;
  cin>>numar;
  if (estePrim(numar))
    cout<<numar<<" este prim.\n";
  else
    cout<<numar<<" nu este prim.\n";
  return 0;
}
```

8) Scrieți o funcție care returnează oglinditul unui număr.

```
int oglindit(int numar)
{
  int og=0;
  int cifra;
  while (numar>0)
  {
    cifra=numar%10;
    og=og*10+cifra;
    numar=numar/10;
  }
  return og;
}
```

9) Alte exemple de funcții

- subprogram care intoarce valoarea sumei cifrelor printr-un parametru

```
void sum_cif(int numar, int & suma)
{
  suma=0;
  while (numar>0)
  {
    suma=suma+numar%10;
    numar=numar/10;
  }
}
```

- subprogram care returneaza o valoare

```
int sum_cif(int numar)
{
 int suma=0;
 while (numar>0)
  {
    suma=suma+numar%10;
    numar=numar/10;
  }
 return suma;
}
```

10) Pentru un şir de n numere naturale citite de la tastatură determinaţi câte din ele sunt câte impare, câte prime şi câte cuburi perfecte.

Rezolvare:

```
#include <stdio.h>
#include <math.h>

long int este_prim(long int m)
{
	for (long int i=2; i<=sqrt(m); i++)
		if (m%i==0) return 0;
	return 1;
}

int este_cub_perfect(long double m)
{
	for (long int i=1; i<=m; i++)
		if (i*i*i==m) return 1;
	return 0;
}

void main()
{
	int n, ncub, nprime, nimp, i; long int a;

	printf("\nDati n:"); scanf("%d",&n); nimp=0; ncub=0; nprime=0;
	for (i=0; i<n; i++)
	{
		printf("Dati un numar:"); scanf("%ld",&a);
		if (a % 2 == 1) nimp++;
		if (este_cub_perfect(a)) ncub++; if (este_prim(a)) nprime++;
	}
	printf("Numarul elementelor impare  = %d\n", nimp);
	printf("Numarul elementelor prime   = %d\n", nprime);
	printf("Numarul elementelor cuburi  = %d\n", ncub);
}
```

11) Pentru un şir de n numere citite de la tastatură determinaţi numărul cel mai mare din şir, cel mai mic număr negativ din şir.

Rezolvare:

```
#include <stdio.h>

long int suma_cifre(long int m)
{
	long int s=0; while (m!=0)
	{ s += m % 10; m/=10; }
	return s;
```

```
}

int main()
{
	int i, n;
	float a, max, min_neg; printf("\nDati n:"); scanf("%d",&n);
	printf("\nDati un numar: "); scanf("%f",&a); max=a;
	min_neg=a;
	for (i=1; i<n; i++)
	{
		printf("\nDati un numar: "); scanf("%f",&a);
		if (a>max) max=a;
		if ((a<min_neg) && (a<0)) min_neg=a;
	}
	printf("\nMaximul este: %0.2f",max);
	if (min_neg<0)
		printf("\nMinimul negativ este: %0.2f",min_neg);
	else
		printf("\nNu exista un numar negativ in sir")
	return 0;
}
```

1.7. Șiruri de caractere de tip String

De foarte multe ori este nevoie ca să lucrezi cu nume de persoane, titluri de cărți, cuvinte de tot soiul, propoziții etc., adică practic cu șiruri de caractere. Există două variante de a lucra cu șiruri de caractere în C++:

- tablouri (vectori) de caractere;
- obiecte din clasa String.

Despre vectori de caractere vom învăța mai târziu, după ce vom învăța despre tablouri, în general. Vei vedea că au niște dezavantaje, pentru că operațiile de atribuire, comparare, concatenare (alipire) și altele sunt mai complicate, necesitând niște funcții speciale, definite în biblioteca **string.h (sau cstring)**. Fiind o chestie mai tehnică, specifică subsetului C din limbajul C++, normal că explicațiile sunt mai complicate.
Dacă, în schimb, vom folosi obiecte din clasa string (definită în biblioteca **string**), lucrurile vor fi mult mai simple, chiar dacă, în spate, conceptele de clasă, metodă și alte lucruri țin de

programarea orientată pe obiecte, care este, de fapt, mai complicată.
Dar pentru început, nu ne interesează aspectele de POO, ci cum să lucrăm simplu cu cuvinte, propoziții și să le folosim în programele noastre atractive.

Pentru a declara o variabilă de tip string, va trebui să includem biblioteca **string**. Constanteled e tip string se scriu între ghilimele. Apoi putem face atribuiri, comparații cu operatorii folosiți la numere, putem citi și putem scrie șiruri de caractere cu cin și cout. De asemenea, putem accesa caracterul de pe o anumită poziție și sunt câteva funcții speciale, numite metode, care ne ajută în a lucra cu șiruri de caractere.
De exemplu, în următorul program vom scrie o funcție care verifică dacă un prenume de persoană (în limba română) este femeie, considerând că numele de femei se termină cu „a”, dar sunt câteva excepții:

- Mihnea, Luca, Mircea, Toma și Horia sunt nume de bărbați, deși se termină cu „a”;
- Alice, Beatrice și Carmen sunt nume de femei, chiar dacă nu se termină cu „a”.

```
#include <string>
#include <iostream>

using namespace std;

bool estePrenumeDeFata(string p)
{
    if (p=="Alice" || p=="Beatrice" || p=="Carmen") return true;
    if (p=="Mircea" || p=="Mihnea" || p=="Toma" || p=="Luca") return false;
    int lung=p.length(); // numarul de caractere ale lui p, numerotate de la 0
la lung-1
    if (p[lung-1]=='a') return true; else return false;
}

int main()
{

    string prenume1, prenume2;
    int varsta1, varsta2;

    cout<<"Cum te numesti? ";
    cin>>prenume1;
    cout<<"Cati ani ai? ";
```

```
    cin>>varsta1;
    cout<<"Cine e prietenul tau sau prietena ta? ";
    cin>>prenume2;
    cout<<"Cati ani are? ";
    cin>>varsta2;

    if (estePrenumeDeFata(prenume1))
        cout<<"Tu esti fata.\n";
    else
        cout<<"Tu esti baiat.\n";
    if (estePrenumeDeFata(prenume2))
        cout<<prenume2<<" este prietena ta.\n";
    else
        cout<<prenume2<<" este prietenul tau.\n";
    if (varsta1<varsta2)
        cout<<"Tu este mai mare.\n";
    else if (varsta1>varsta2)
        cout<<"Tu esti mai mic(a).\n";
    else
        cout<<"Aveti aceeasi varsta.\n";
    return 0;
}
```

```
Cum te numesti? Costel
Cati ani ai? 15
Cine e prietenul tau sau prietena ta? Maria
Cati ani are? 14
Tu esti baiat.
Maria este prietena ta.
Tu esti mai mic(a).
```

1.8. Probleme recapitulative pentru capitolul 1

1) Viteza unui automobil este de v km/h. Exprimaţi această viteză în m/s.

2) Un muncitor lucrează n zile pentru a termina o lucrare. Să se determine câte zile sunt necesare pentru a termina aceeaşi lucrare o echipă de m muncitori.

3) Două echipe de muncitori au în componenţa lor m_1, respectiv m_2 muncitori. Prima echipă termină o lucrare în z_1 zile, iar cea de a doua în z_2. Să se determine:

- z_2 în funcţie de m_1, z_1, m_2;
- m_2 în funcţie de m_1, z_1, z_2.

4) O carte are n foi şi r rânduri pe fiecare pagină. Câte rânduri are cartea?

5) Să se determine aria unui trapez când se cunosc lungimile bazelor şi a înălţimii.

6) Să se determine lungimea unui cerc în funcţie de aria sa.

7) Să se determine perimetrul şi aria unui triunghi echilateral înscris într-un cerc de diametru d.

8) Să se determine cel de al treilea unghi al unui triunghi când se cunosc celelalte două.

9) Să se determine aria pătratului circumscris unui cerc, cunoscând aria pătratului înscris în cerc.

10) Catetele unui triunghi dreptunghic au lungimile a şi b. Să se determine ipotenuza, perimetrul şi aria sa.

11) Un punct material se află la distanţa x_0 de origine, la momentul iniţial t_0, când începe să se mişte rectiliniu uniform. Știind că la momentul t se află la distanţa x faţă de origine, să se determine viteza v de mişcare a punctului material, la momentul t.

12) Scrieţi funcţii care să calculeze aria unui cerc în funcţie de: a) raza cercului; b) diametrul cercului; c) lungimea cercului; d) latura triunghiului echilateral înscris în cerc; e) latura pătratului înscris în cerc.

13) Să se rezolve un triunghi (adică să se determine elementele sale necunoscute (lungimi de laturi sau măsuri de unghiuri)), când se cunosc:

- lungimile celor trei laturi;
- lungimea a două laturi şi măsura unghiului dintre ele;
- lungimea unei laturi şi măsurile unghiurilor adiacente ei

14) Scrieţi o funcţie care să transforme un unghi exprimat în radiani în valoarea sa exprimată în grade şi una care să facă transformarea inversă.

15) Scrieţi o funcţie care să determine diferenţa dintre două momente de timp, date prin ore, minute şi secunde.

16) Aceeaşi problemă ca şi cea anterioară, dar cu citiri/afişări colorate, în diferite poziţii ale ecranului text.

17) Să se determine dacă un număr a este divizibil cu toate numerele b_1, b_2 și b_3.

18) Să se calculeze suma: $S=1-1\times3+1\times3\times5-...\pm1\times3\times5\times...\times(2n-1)$.

19) Scrieţi o funcţie int power(int x, int n) care ridică pe x la puterea n folosind instrucțiunea for.

20) Elaboraţi un program care să tipărească tabela de temperaturi Fahrenheit şi echivalenţele lor în centigrade sau grade Celsius, folosind formula: C=(5/9)*(F-32), între valorile extreme 0 şi 300 grade Fahrenheit.

21) Elaboraţi un program care să tipărească tabela corespunzătoare Celsius-Fahrenheit.

22) Scrieţi un program care să calculeze factorialul unui număr întreg n.

23) Venitul unei societăţi comerciale este de venit milioane lei, iar cheltuielile sunt de chelt lei. Să se precizeze dacă profitul societăţii este mai mare sau nu decât o valoare constantă profit_minim = 500 (milioane lei).

24) Într-o magazie sunt următoarele produse: fasole, cartofi şi roşii în cantităţile f, c, r. Un kilogram de fasole costă cf lei, unul de cartofi cc lei. Să se determine costul unui kilogram de roşii, ştiind că toate legumele din magazie costă C lei.

25) Scrieţi un program care să rezolve ecuaţia de gradul I ax+b=0.

26) Scrieţi un program care să rezolve ecuaţia de gradul II $ax^2+bx+c=0$, inclusiv pentru a=0 sau când ecuaţia admite soluţii complexe, nereale.

27) Care este valoarea variabilei p după efectuarea următoarelor operaţii?
p=i=1; while (i<7) i++; p*=i;
p=1; i=7; while (i>1) {p*=i; i-=1;}

28) Ce este greşit în secvenţa de instrucţiuni de mai jos:
if (n<5) x=while; else i:=i+1; while (p<4) k++;

29) Fie a, b, c trei numere reale. Scrieţi o funcţie care să verifice dacă a, b pot reprezenta lungimile laturilor unui dreptunghi, respectiv c să fie diagonală în acel dreptunghi.

30) Ce execută următorul program C ? Rescrieţi-l folosind instrucţiunea for.

```
#include <stdio.h>
int main()
{
        long nc=0;
```

```
        while (getchar() != EOF) ++nc; printf("%ld\
n",nc); return 0;
}
```

31) Scrieţi un program care să numere spaţiile, tab-urile şi new-line-urile (trecerile la un nou rând) date de la tastatură.

32) Scrieţi un program care să copieze intrarea în ieşire, înlocuind fiecare şir de unul sau mai multe spaţii cu un singur spaţiu.

33) Scrieţi un program care să înlocuiască fiecare tab printr-o secvenţă de trei spaţii.

34) Scrieţi un program care să determine cel mai mare divizor comun a două numere întregi, prin algoritmul clasic al lui Euclid (cu împărţiri repetate).

35) Să se determine dacă două numere sunt prime între ele sau nu.

36) Să se scrie programe pentru a deplasa o bilă pe ecran, de la stânga la dreapta şi de la dreapta la stânga, până la acţionarea unei taste.

37) Scrieţi un program care să citească mai multe numere întregi, până la întâlnirea lui zero, şi să determine cel mai mic număr şi cel mai mare număr citit.

38) Scrieţi un program care să calculeze valoarea unui polinom într-un punct dat. Polinomul este dat prin coeficienţii săi. Nu se vor utiliza tablouri!

39) Să se scrie un program care să citească două numere reale a şi b. Apoi să pună utilizatorului o întrebare: *Ce doriţi să calculăm ? Media aritmetică (1) sau geometrică (2)?*. Dacă se va răspunde prin 1, se va calcula şi afişa media aritmetică, iar pentru 2 media geometrică (numai dacă numerele sunt pozitive !, iar de nu, se va afişa ‘*eroare !*’). Dacă nu se răspunde prin 1 sau 2 se va produce un sunet în difuzor.

40) Să se scrie un program care să citească trei numere reale a, b şi c, apoi să pună o întrebare de genul: *Ce doriţi să calculăm? Aria sau perimetrul?*. Dacă se va răspunde prin ‘*aria*’ atunci se va calcula şi afişa aria, altfel perimetrul. Ce se va întîmpla dacă se va răspunde prin ‘*arie*’ ?

41) Să se scrie un program care să citească un număr întreg, iar dacă acest număr este 1 să afişeze *luni*, dacă este 2 să afişeze *marţi*, ... 7 - *duminică*, iar dacă nu este cuprins între 1 şi 7 să afişeze cuvîntul *eroare*. Să se scrie programul folosind: a) instrucţiunea if; b) instrucţiunea switch.

42) Să se scrie un program care să calculeze aria unui cerc în funcţie de diametru, dacă răspunsul la întrebarea: ‘*Aria în funcţie de diametru (1) sau lungime (2)?*” este fie caracterul ‘1’, fie caracterul ‘D’, sau să calculeze aria cercului în funcţie de lungime, dacă răspunsul este ‘2’ sau ‘L’, iar dacă răspunsul este diferit de aceste patru posibilităţi să se producă un sunet.

43) Să se precizeze dacă un triunghi cu lungimile celor trei laturi a, b şi c este sau nu isoscel. În caz afirmativ, să se producă un sunet şi să se afişeze *Tra-la-la* în mijlocul ecranului, altfel să se rezolve ecuaţia $ax^2+bx+c=0$, unde a şi b sunt lungimile primelor două laturi ale triunghiului, iar c este semiperimetrul său.

44) Să se realizeze următorul joc: o bilă se mişcă pe ecran ca pe o masă de biliard (vezi problema 2). O paletă de lungime 9 (caractere), situată pe ultima linie a ecranului, se deplasează stânga-dreapta cu ajutorul a două taste (vezi problema 3). Când bila loveşte paleta, în difuzor se aude un sunet mai lung, iar când ea loveşte pereţii laterali, un alt sunet, mai scurt. Jocul se opreşte când se apasă o anumită tastă de stop.

45) Se citesc de la tastatură caractere, până la întâlnirea a două caractere care sunt identice. Să se determine câte caractere sunt cifre dintre cele citite.

46) De la tastatură se introduce o listă de numere întregi. Se cere să se afişeze valoarea maximă depistată în listă. Dimensiunea listei este precizată înainte de a fi introduse elementele ce o compun.

47) Să se scrie un program care să deseneze un dreptunghi umplut cu o anumită culoare (o cutie). Dreptunghiul se va specifica prin coordonatele colţurilor stânga-jos şi lungimile laturilor, care, alături de culoare, vor fi introduse de la tastatură.

48) Să se scrie un program care să deseneze pe ecran dreptunghiuri de dimensiuni aleatoare, în poziţii aleatoare şi de culori aleatoare pe ecran, pînă se acţionează o tastă numerică.

49) Realizaţi un program care să afişeze numele anotimpului corespunzător unui număr întreg citit de la tastatură (1..4). Scrieţi două variante de program: folosind if şi folosind switch.

50) Scrieţi un program care să pună întrebarea: “*În ce an s-a născut Eminescu ?*” şi. dacă răspunsul este corect (*1850*), atunci să afişeze “*Foarte bine*”, altfel un text corespunzător. Folosiţi instrucţiunea switch.

51) Se citeşte un şir de numere întregi pâna la întâlnirea numărului 0. Să se calculeze media aritmetică a numerelor din şir.

52) Se citesc 3 numere naturale n, p şi k, apoi un şir de n numere naturale. Câte dintre acestea, împărţite la p dau restul k ?

53) Să se calculeze produsul a două numere naturale prin adunări repetate.

54) Efectuaţi împărţirea întreagă a două numere, fără a utiliza operatorii / şi %, ci doar scăderi repetate.

55) Se citeşte un număr natural. Câte cifre conţine ?

56) Un număr se numeşte “palindrom” dacă citit invers este acelaşi număr. Să se verifice dacă un număr este sau nu palindrom.

57) Să se afişeze toate numerele prime mai mici sau egale cu un numar m dat.

58) Să se afişeze primele n numere prime care au suma cifrelor m.

59) Să se afişeze toate numerele de 3 cifre care, citite invers, sunt tot numere prime.

60) Calculaţi şi afişaţi suma: 1/(12) + 1/(23) + 1/(34) + ... + 1/(n(n+1)).

61) Să se verifice dacă un număr natural este sau nu pătrat perfect.

62) Să se scrie un program care să rezolve câte o ecuatie de gradul II, până utilizatorul programului nu mai vrea acest lucru.

63) Să se verifice dacă un număr natural este sau nu cub perfect.

64) Să se afişeze toate numerele de forma a^2+b^3, cu 1<=a<=5 şi 1<=b<=5.

65) Să se determine toate triunghiurile diferite cu laturi numere întregi pozitive şi perimetru p.

66) Să se listeze toate numerele n, a căror sumă a cifrelor este divizibilă prin 5.

67) Să se transforme un număr din baza 10 în baza p<10. Să se transforme un număr din baza p<10 în baza 10.

68) Să se transforme un număr din baza p în baza q, unde p, q 10.

69) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze soluţiile tuturor ecuaţiilor de gradul I ax+b=0, unde a şi b sunt toate perechile de numere citite consecutiv, în care b este divizibil prin a.

70) Se citesc numere naturale până la introducerea unui număr real. Să se calculeze suma S a tuturor numerelor citite, precum şi câtul şi restul împărţirii lui S la suma cifrelor lui S.

71) Se citesc numere naturale până la întâlnirea numărului 12. Să se afişeze toate tripletele de numere citite consecutiv, în care al treilea număr este restul împărţirii primului la al doilea.

72) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze toate tripletele de numere citite consecutiv, în care al treilea număr este media aritmetică (geometrică) dintre primul şi al doilea.

73) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze toate perechile de numere citite consecutiv, cu proprietatea că al doilea număr este egal cu suma cifrelor primului număr.

74) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze toate perechile de numere citite consecutiv, cu proprietatea că al doilea număr reprezintă restul împărţirii primului număr la suma cifrelor sale.

75) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze toate perechile de numere citite consecutiv, cu proprietatea că al doilea număr reprezintă numărul de apariţii ale cifrei 3 în pătratul primului.

76) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze toate perechile de numere citite consecutiv, cu proprietatea că al doilea număr reprezintă pătratul numărului de apariţii ale cifrei *1* în primul.

77) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze toate tripletele de numere citite consecutiv, cu proprietatea că al treilea număr este suma dintre primul şi al doilea.

78) Se citesc numere reale până la întâlnirea numărului 0. Să se afişeze toate tripletele de numere citite consecutiv, cu proprietatea că ele pot reprezenta laturile unui triunghi.

79) Se citesc numere reale până la întâlnirea numărului 0. Să se afişeze toate tripletele de numere citite consecutiv, cu proprietatea că ele pot reprezenta laturile unui triunghi isoscel.

80) Se citesc numere reale până la întâlnirea numărului 0. Să se afişeze toate tripletele de numere citite consecutiv, cu proprietatea că ele pot reprezenta laturile unui triunghi dreptunghic.

81) Se citesc numere reale până la întâlnirea numărului 0. Să se afişeze soluţiile tututor ecuaţiilor de gradul 2 $ax^2+bx+c=0$, unde a, b şi c sunt toate

tripletele de numere citite consecutiv, în care $b^2-4ac=0$.

82) Se citesc numere reale până la întâlnirea numărului 0. Să se afişeze soluţiile tututor ecuaţiilor de gradul II $ax^2+bx+c=0$, unde a, b şi c sunt toate tripletele de numere citite consecutiv, în care $b^2-4ac>0$.

83) Se citesc numere naturale până la întâlnirea numărului 0. Să se afişeze toate perechile de numere (n1,n2) citite consecutiv, cu proprietatea că n1>n2 şi suma cifrelor lui n1 este mai mică decât suma cifrelor lui n2.

84) Să se găsească toate reprezentările posibile ale numărului natural n ca sumă de numere naturale consecutive.

85) Să se calculeze determinantul unei matrice pătratice de ordinul doi şi a unei matrice pătratice de ordinul trei.

86) La un concurs sportiv s-au înscris patru echipe A, B, C şi D. La sfârşitul tuturor probelor cele patru echipe au obţinut respectiv X, Y, Z şi T puncte. Să se afişeze valoarea de adevăr a propoziţiei: "*Echipa A a obţinut cel mai mare punctaj.*"

87) Scrieţi un program care să se rezolve ecuaţia de gradul III: $ax^3+bx^2+cx+d=0$, a, b, c, d numere reale, a0, folosind *metoda lui Cardano*.

88) Generaţi trei numere reale din intervalul [0,1) şi apoi calculaţi şi afişaţi media lor aritmetică.

89) Prima defectare a unui calculator survine după o durată de funcţionare specificată prin trei numere întregi: h ore, m minute şi s secunde. Scrieţi un program care să determine durata de funcţionare în secunde.

90) Scrieţi un program care să rezolve un sistem de două ecuaţii liniare cu două necunoscute: $a_1x+b_1=c_1$, $a_2x+b_2=c_2$.

91) Scrieţi un program care să determine elementele mulţimii $\{x|x\in R, \cos(n\times\arccos(x))=0\}$.

92) Calculaţi media armonică a trei numere reale x, y şi z.

93) Scrieţi funcţii şi apoi dezvoltaţi programe C pentru rezolvarea următoarelor probleme:

94) Fie A(x1,x2) şi B(x2,y2) două puncte din planul euclidian. Afişaţi: distanţa euclidiană de la A la B; coordonatele mijlocului segmentului |AB|.

95) Fie A(x1,y1), B(x2,y2) şi C(x3,y3) cele trei puncte din planul euclidian ce reprezintă colţurile unui triunghi ABC. Să se determine: coordonatele mijloacelor laturilor; lungimile laturilor; lungimile medianelor.

96) Pentru datele de la punctul anterior, să se determine coordonatele cercului înscris în triunghi şi a celui circumscris triunghiului, precum şi razele acestor cercuri.

97) Să se determine coeficienţii a, b şi c ai dreptei ax+by+c=0, care trece prin punctele P şi Q de coordonate date.

98) Scrieţi un program care să verifice dacă trei puncte din planul euclidian, date prin coordonatele lor (x1,y1), (x2,y2) şi (x3,y3) sunt sau nu coliniare.

99) Să se verifice dacă trei drepte, date prin ecuaţiile lor de forma ax+by+c=0, sunt sau nu concurente.

100) Să se calculeze distanţa de la un punct $P(x_p,y_p)$ la dreapta ax+by+c=0.

101) Dându-se trei numere reale a, b şi c, să se rezolve ecuaţia $ax^4+bx^2+c=0$, a0.

102) Scrieţi un program C care să studieze natura şi semnul soluţiilor unei ecuaţii de gradul doi.

103) Se citeşte un număr natural n şi se cere să se afişeze descompunerea lui în factori primi.

104) Să se afişeze toate modurile în care se poate descompune un număr natural n ca sumă de numere naturale consecutive.

105) Pentru un număr n natural citit de la tastatură să se determine suma divizorilor săi pozitivi, care sunt numere prime.

106) Pentru un număr n natural citit de la tastatură să se calculeze suma numerelor prime mai mici decât n.

107) Pentru un şir de n numere citite de la tastatură, care nu sunt divizibile prin 10, determinaţi numărul a cărui oglindit este cel mai mic.

108) Pentru un şir de n numere citite de la tastatură, determinaţi câte din ele sunt identice cu oglinditele lor şi câte au suma cifrelor un pătrat perfect.

109) Să se simuleze mişcarea unei bile de jos în sus, pe coloana C a ecranului.

110) Să se simuleze mişcarea unei bile de la stânga la dreapta şi apoi de la dreapta la stânga, în mod repetat, pe linia L a ecranului, până la acţionarea unei taste.

111) Să se simuleze pe ecran mişcarea a două bile de la stânga la dreapta şi invers şi de sus în jos şi invers, simultan, până la acţionarea unei taste.

112) Se citeşte numărul natural n de la tastatură. Să se calculeze, folosind toate instrucţiunile repetitive cunoscute, expresiile:

- $S = 1 + 1\times2 + 1\times2\times3 + ... 1\times2\times...\times n$;
- $S = 1 - 1\times3 + 1\times3\times5 - 1\times3\times5\times7 + ...$ (n termeni)
- $S = 1^2 - 1/(1\times2) + 3^2 - 1/(1\times2\times3\times4) + 5^2 - 1/(1\times2\times3\times4\times5\times6) + ...$ (n termeni)
- $S = 1^2 - 1\times4^2 + 1\times4\times7^2 - 1\times4\times7\times10^2 + ...$ (n termeni)

Capitolul 2. Aplicații cu tablouri unidimensionale și structuri

2.1. O bibliotecă cu funcții de intrare și ieșire foarte utile

```
#include <pcr.h>

int main()
{
  int i;
  forc(i,2,9)
    write(i," ");
  writeln("");
  ford(i,11,3)
    write(i," ");

  return 0;
}

#define forc(x,e1,e2) for(x=e1; x<=e2; x++)
#define ford(x,e1,e2) for(x=e1; x>=e2; x--)


#include <iostream>

using namespace std;

void read(unsigned int & x)
{
  cin>>x; cin.ignore();
}

void read(float & x)
{
  cin>>x; cin.ignore();
}

void read(string & x)
{
  getline(cin,x);
}

void read(char & c)
{
  cin>>c; cin.ignore();
}

template <typename T>

void write(T t)
{
  cout << t ;
}

template <typename T>
void writeln(T t)
{
```

```
  cout << t << endl;
}

template<typename T, typename... Args>

void write(T t, Args... args)
{
  cout << t << " " ;
  write(args...) ;
}

template<typename T, typename... Args>

void writeln(T t, Args... args)
{
  cout << t << " " ;
  write(args...) ;
  cout<<"\n";
}
```

2.2. Funcții (partea a II-a)

```
#include <iostream>
#include <pcr.h>

using namespace std;

void pune(int x[100], int n)
{
  for (int i=0;i<n;i++)
    x[i]=i;
}

// sa se interschimbe valorile a doua variabile intregi

void schimba(int &a, int &b)
{
  int aux;
  aux=a;
  a=b;
  b=aux;
}

// sa se scrie o functie care returneaza suma elementelor dintr-un vector x cu n
numere intregi

int suma(int v[100], int n)
{
  int s=0;
  for (int i=0;i<n;i++)
    s+=v[i];
```

```
  return s;
}

// sa se scrie o functie care citeste de la tastatura un vector x cu n numere
intregi
void citeste(int v[100], int & n)
{
  cin>>n;
  for (int i=0;i<n;i++)
    cin>>v[i];
}

int main()
{
  int a[100];

  pune(a,5);
  for (int i=0;i<5;i++)
    cout<<a[i]<<" ";
  int z=suma(a,5);
  writeln(z);
  int n;
  citeste(a,n);
  for (int i=0;i<n;i++)
    cout<<a[i]<<" ";
  /*int x=3,y=8;
  schimba(x,y);
  writeln(x,y); */
  return 0;
}
```

2.3. Pisicile

Cate pisici birmaneze cu varsta > 5 ani sunt intre doua pisici siameze in sir?

```
#include <iostream>
#include <string>

using namespace std;

struct pisica {
  string nume;
  string rasa;
  int varsta;
};

int main()
{
  int i;
  pisica p[51]; // sirul de pisici
  int n; // numarul de pisici
  cout<<"Programul cu pisici:\n";
```

```
  cout<<"Dati numarul de pisici: "; cin>>n; cin.ignore();
  for (i=1;i<=n;i++)
  {
    cout<<"Dati informatiile despre pisica nr. "<<i<<":\n";
    cout<<"- numele: "; cin>>p[i].nume;
    cout<<"- rasa:  "; cin>>p[i].rasa;
    cout<<"- varsta: "; cin>>p[i].varsta; cin.ignore();
  }

  cout<<"Cate pisici birmaneze cu varsta > 5 ani sunt intre doua pisici siameze
in sir?\n";
  int nr=0;
  for (i=2;i<=n-1;i++)
    if (p[i].varsta>5 && p[i].rasa=="birmaneza" && p[i-1].rasa=="siameza"
       && p[i+1].rasa=="siameza")
      nr=nr+1;
  cout<<"Sunt "<<nr<<" asemenea pisici.\n";

  return 0;
}
```

2.4. Elevii

```
#include <iostream>
#define MAX 100

using namespace std;

struct elev {
  char nume[15];
  int varsta;
  int romana;
  int mate;
  float media;
};

int nrElevi;

elev E[MAX];

int main()
{
  cout<<"Dati numarul de elevi: "; cin>>nrElevi;
  int i;
  for (i=1;i<=nrElevi; i++)
  {
    cout<<"Dati datele despre elevul nr. "<<i<<":\n";
    cout<<"numele: "; cin>>E[i].nume;
    cout<<"nota la romana: "; cin>>E[i].romana;
    cout<<"nota la matematica: "; cin>>E[i].mate;
    E[i].media=(E[i].romana+E[i].mate)/2.0;
```

```
  }
  for (i=1;i<=nrElevi; i++)
    cout<<"Elevul nr. "<<i<<" se numeste "<<E[i].nume<<" si a obtinut media
"<<E[i].media<<endl;
  return 0;
}

—------------------------------



#include <iostream>

using namespace std;

int main()
{
  int varsta[51];
  int sex[51];
  int n,i;
  // Se dau n varste de copii si 1=baiat, 2=fata
  // Cati baieti de cel putin 15 ani sunt cuprinsi intre doua fete?
  cout<<"nr de copii="; cin>>n;
  for (i=1;i<=n;i++)
  {
    cout<<"Persoana nr. "<<i<<":\n";
    cout<<"varsta:";
    cin>>varsta[i];
    cout<<"1 = baiat, 2 = fata:";
    cin>>sex[i];
  }
  int numar=0;
  for (i=2;i<=n-1;i++)
    if (sex[i]==1 && varsta[i]>=15 && sex[i-1]==2 && sex[i+1]==2)
      numar++;
  cout<<"Total "<<numar<<" baiati cu varste >=15 ani si cuprinsi intre doua
fete.\n";

  return 0;
}

—---------------------------------
#include <iostream>
#define MAX 100

using namespace std;

struct elev {
  char nume[15];
  int varsta;
  int romana;
  int mate;
```

```
  float media;
};

int nrElevi;

elev E[MAX];

int main()
{
  int i;

  // citirea datelor de intrare
  cout<<"Dati numarul de elevi: "; cin>>nrElevi;
  for (i=1;i<=nrElevi; i++)
  {
    cout<<"Dati datele despre elevul nr. "<<i<<":\n";
    cout<<"numele: "; cin>>E[i].nume;
    cout<<"nota la romana: "; cin>>E[i].romana;
    cout<<"nota la matematica: "; cin>>E[i].mate;
  }

  // calcularea datelor de iesire in functie de datele de intrare
  for (i=1;i<=nrElevi;i++)
    E[i].media=(E[i].romana+E[i].mate)/2.0;

  // afisarea datelor de iesire
  for (i=1;i<=nrElevi; i++)
    cout<<"Elevul nr. "<<i<<" se numeste "<<E[i].nume<<" si a obtinut media
"<<E[i].media<<endl;
  return 0;
}
—------------------------------


#include <iostream>
#define MAX 100

using namespace std;

struct elev {
  char nume[15];
  int varsta;
  int romana;
  int mate;
  float media;
};

void citesteDate(elev &X)
{
  cout<<"nume: "; cin>>X.nume;
  cout<<"varsta: "; cin>>X.varsta;
  cout<<"nota la romana: "; cin>>X.romana;
```

```
  cout<<"nota la matematica: "; cin>>X.mate;
}

int nrElevi;

elev E[MAX];

int main()
{
  int i;

  // citirea datelor de intrare
  cout<<"Dati numarul de elevi: "; cin>>nrElevi;
  for (i=1;i<=nrElevi; i++)
  {
    cout<<"Elevul nr. "<<i<<":\n";
    citesteDate(E[i]);
  }

  // calcularea datelor de iesire in functie de datele de intrare
  for (i=1;i<=nrElevi;i++)
    E[i].media=(E[i].romana+E[i].mate)/2.0;

  // afisarea datelor de iesire
  for (i=1;i<=nrElevi; i++)
    cout<<"Elevul nr. "<<i<<" se numeste "<<E[i].nume<<" si a obtinut media
"<<E[i].media<<endl;
  return 0;
}
```

```
#include <pcr.h>
#include <stdio.h>

#include <string>
#include <algorithm>

#define MAX 100

struct elev {
  string nume;
  string prenume;
  char sex;
  unsigned int varsta;
  float media;
  bool bursier;
};

elev Elev[MAX];
unsigned int nrElevi;
```

```
void meniu()
{
  writeln("");
  writeln("Program gestionare elevi:");
  writeln("1 - citire date elevi de la tastatura");
  writeln("2 - sortare elevi dupa medie, descrescator");
  writeln("3 - sortare elevi alfabetic");
  writeln("4 - afisare elevi major, apoi minori");
  writeln("5 - afisare elevi dupa sex (intai fete, apoi baieti)");
  writeln("6 - incarcare date elevi din fisier");
  writeln("7 - salvare date elevi in fisier");
  writeln("0 - oprire program");
}

void citire()
{
  writeln("Citire date elevi");
  write("Dati numarul de elevi:","n=");
  read(nrElevi);
  for (unsigned int i=1; i<=nrElevi; i++)
  {
    writeln("Dati datele pentru elevul nr. ", i);
    write(" - numele: "); read(Elev[i].nume);
    write(" - prenumele: "); read(Elev[i].prenume);
    write(" - sex (m/f): "); read(Elev[i].sex);
    write(" - varsta: "); read(Elev[i].varsta);
    write(" - media: "); read(Elev[i].media);
  }
}

void afisareDateElev(int i)
{
  write(Elev[i].nume, Elev[i].prenume, "(",Elev[i].sex,")", Elev[i].varsta,
"ani,",Elev[i].media);
  if (Elev[i].media>=8.50) writeln("-> bursier"); else writeln("");
}

void afisareMajoriMinori()
{
  write("Afisare elevi majori, apoi minor\n");
  write("Elevi majori:\n");
  for (unsigned int i=1; i<=nrElevi; i++)
  {
    if (Elev[i].varsta>=18) afisareDateElev(i);
  }
  write("Elevi minori:\n");
  for(unsigned int i=1; i<=nrElevi; i++)
    if (Elev[i].varsta<18) afisareDateElev(i);
}

void afisareFeteBaieti()
{
```

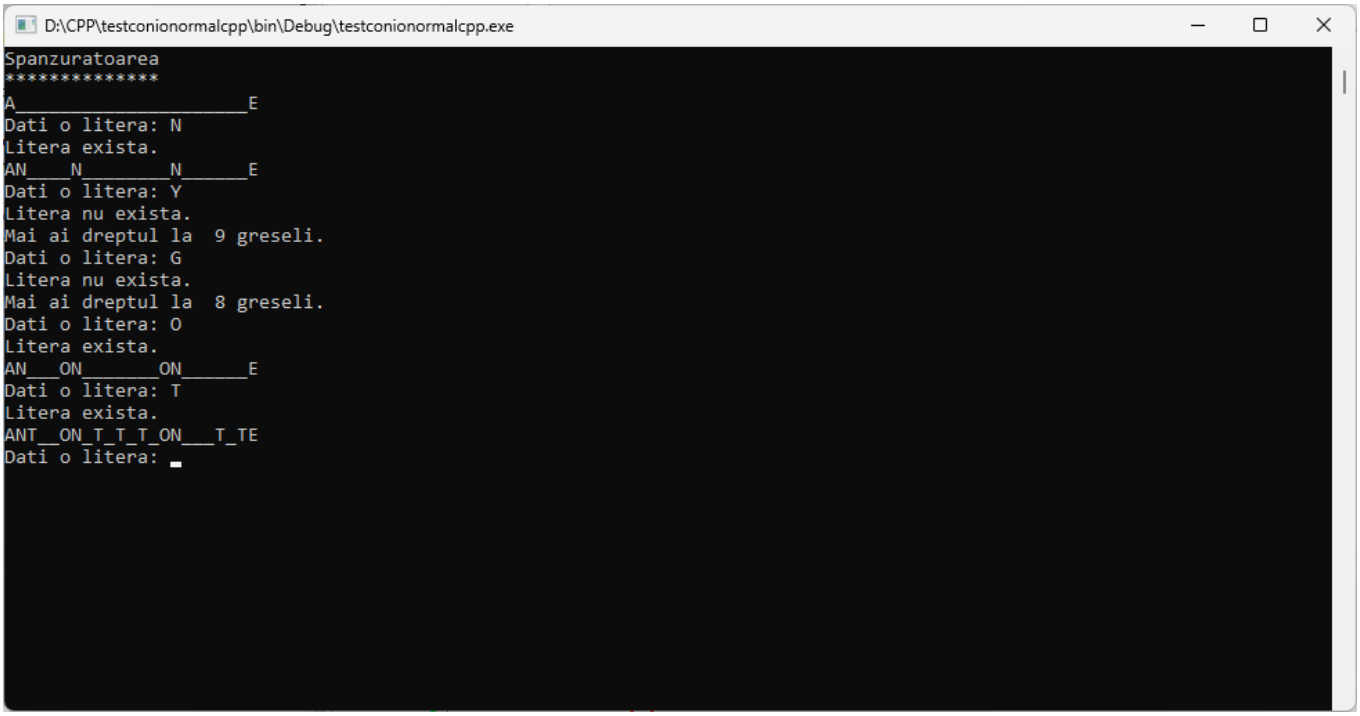
```
  write("Afisare fete, apoi baieti\n");
  write("Fete:\n");
  for(unsigned int i=1; i<=nrElevi; i++)
  {
    if (Elev[i].sex == 'f') afisareDateElev(i);
  }
  write("Baieti:\n");
  for(unsigned int i=1; i<=nrElevi; i++)
  {
    if (Elev[i].sex == 'm') afisareDateElev(i);
  }
}

bool alfabet(elev E1, elev E2)
{
  if (E1.nume == E2.nume)
  {
    return E1.prenume <= E2.prenume;
  }
  return E1.nume < E2.nume;
}

void sortareAlfabeticDupaNume()
{
  sort(Elev+1, Elev+nrElevi+1, alfabet);
  write("Elevii in ordinea alfabetica:\n");
  for(unsigned int i=1; i<=nrElevi; i++)
    afisareDateElev(i);
}

bool medieDescr(elev E1, elev E2)
{
  return E1.media >= E2.media;
}

void sortareMedieDescr()
{
  sort(Elev+1,Elev+nrElevi+1, medieDescr);
  write("Elevii descrescator dupa medie;\n");
  for(unsigned int i=1; i<=nrElevi; i++)
    afisareDateElev(i);
}

void afisareTotiElevii()
{
  for (unsigned int i=1; i<=nrElevi; i++)
    afisareDateElev(i);
  writeln("Total",nrElevi,"elevi.");
}

void salvareBin()
{
```

```
  FILE * fisier;
  if ((fisier = fopen("dateelevi.pcr","wb")) == NULL)
    {
      printf("Eroare la deschiderea fisierului dateelevi.pcr pentru scriere in
el.");
      exit(1);
    }
  fwrite(&nrElevi, sizeof(nrElevi), 1, fisier);
  unsigned int lungimeNume, lungimePrenume;
  for (unsigned int i=1;i<=nrElevi;i++)
    {
      // deoarece numele si prenumele pot fi de lungimi variabile de la o
persoana la alta
      // vom salva in fisier si cat de lungi sunt ele, pentru a putea le putea
restaura bine
      lungimeNume=Elev[i].nume.length();
      fwrite(&lungimeNume,sizeof(unsigned int), 1, fisier);
      fwrite(&Elev[i].nume[0], sizeof(char), lungimeNume, fisier);

      lungimePrenume=Elev[i].prenume.length();
      fwrite(&lungimePrenume,sizeof(unsigned int), 1, fisier);
      fwrite(&Elev[i].prenume[0], sizeof(char), lungimePrenume, fisier);

      // campurile acestea nu ridica probleme
      fwrite(&Elev[i].sex, sizeof(char), 1, fisier);
      fwrite(&Elev[i].varsta, sizeof(unsigned int), 1, fisier);
      fwrite(&Elev[i].media, sizeof(float), 1, fisier);
      fwrite(&Elev[i].bursier, sizeof(bool), 1, fisier);
    }
 fclose(fisier);
 writeln("S-au salvat in fisier acesti elevi:");
 afisareTotiElevii();
}

void incarcareBin()
{
  FILE * fisier;
  if ((fisier = fopen("dateelevi.pcr","rb")) == NULL)
    {
      printf("Eroare la deschiderea fisierului dateelevi.pcr pentru incarcare
din el.");
      exit(1);
    }
  fread(&nrElevi, sizeof(nrElevi), 1 , fisier);
  unsigned int lungimeNume, lungimePrenume;
  char buffer[MAX];
  for (unsigned int i=1; i<=nrElevi; i++)
    {
      // deoarece numele si prenumele pot fi de lungimi variabile de la o
persoana la alta
      // vom prelua din fisier cat de lungi sunt ele, pentru a putea le putea
restaura bine
```

```
      fread(&lungimeNume, sizeof(unsigned int), 1, fisier);
      fread(buffer, sizeof(char), lungimeNume, fisier);
      Elev[i].nume=string(buffer,lungimeNume);

      fread(&lungimePrenume, sizeof(unsigned int), 1, fisier);
      fread(buffer, sizeof(char), lungimePrenume, fisier);
      Elev[i].prenume=string(buffer,lungimePrenume);

      // campurile acestea nu ridica probleme
      fread(&Elev[i].sex, sizeof(char), 1, fisier);
      fread(&Elev[i].varsta, sizeof(unsigned int), 1, fisier);
      fread(&Elev[i].media, sizeof(float), 1, fisier);
      fread(&Elev[i].bursier, sizeof(bool), 1, fisier);
    }
  fclose(fisier);
  writeln("S-au restaurat din fisier acesti elevi:");
  afisareTotiElevii();
}

int main()
{
  unsigned int comanda;

  do {
    meniu();
    //cout<<sizeof(string);
    write("Ce doresti? ");
    read(comanda);
    writeln("");
    switch(comanda)
    {
      case 1: citire(); break;
      case 2: sortareMedieDescr(); break;
      case 3: sortareAlfabeticDupaNume(); break;
      case 4: afisareMajoriMinori(); break;
      case 5: afisareFeteBaieti(); break;
      case 6: incarcareBin(); break;
      case 7: salvareBin(); break;
    }
  } while (comanda!=0);
  return 0;
}
```

Capitolul 3. Aplicații la șiruri de caractere

3.1. Jocul Spânzurătoarea (fără grafică)



Acest program implementează jocul "Spânzurătoarea", în care utilizatorul trebuie să ghicească un cuvânt, introducând litere, iar dacă literele introduse apar în cuvântul de ghicit, atunci acestea sunt afișate, altfel se adaugă o greșeală la contorul de greșeli și se afișează câte greșeli mai poate face utilizatorul. Dacă utilizatorul ghicește cuvântul, atunci câștigă jocul, altfel îl pierde.

În ceea ce privește implementarea programului:

- Se începe prin definirea a două variabile de tip string: CuvantDeGhicit și CuvantEcran. CuvantDeGhicit este cuvântul pe care utilizatorul trebuie să-l ghicească, iar CuvantEcran este o versiune a acestui cuvânt cu caracterele necunoscute reprezentate prin '_'.
- Se deschide un fișier "cuvinte.txt" care conține o listă de cuvinte, numărul lor fiind stocat la începutul fișierului. Se generează un număr aleatoriu între 1 și numărul de cuvinte din fișier și se citesc cuvintele până la cuvântul cu numărul ales aleatoriu. Astfel, CuvantDeGhicit va fi cuvântul care trebuie ghicit în joc.
- Se afișează titlul jocului și cuvântul cu caractere necunoscute.
- Se definește variabila nrGreseli, care va fi folosită pentru a număra greșelile utilizatorului, și variabila litera, care va fi folosită pentru a citi literele introduse de utilizator.
- Se pornește o buclă care rulează până când utilizatorul ghicește cuvântul sau face prea multe greșeli. În acest loop se cere utilizatorului să introducă o literă și se verifică dacă aceasta apare în cuvântul de ghicit. Dacă apare, atunci se afișează un mesaj corespunzător și se actualizează cuvântul afișat cu litera ghicită. Dacă litera nu apare, se adaugă o greșeală la contorul de greșeli și se afișează un mesaj corespunzător. Dacă utilizatorul a făcut prea multe greșeli, atunci jocul se termină și se afișează un mesaj corespunzător.

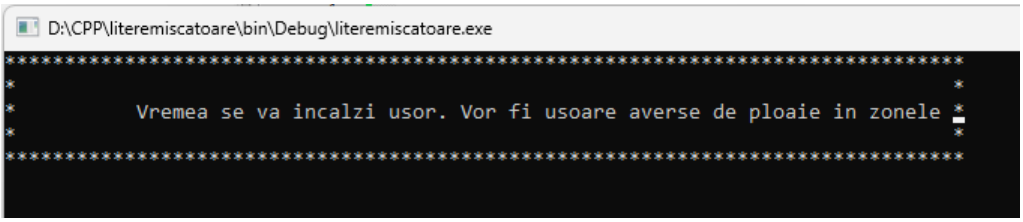
- După ce bucla se termină, se verifică dacă utilizatorul a ghicit cuvântul și se afișează un mesaj corespunzător.

☞ Astfel, acest program implementează un joc simplu de ghicit cuvinte folosind șiruri de caractere și un fișier de intrare pentru alegerea cuvântului de ghicit.

- string CuvantDeGhicit, CuvantEcran;: se declară două obiecte de tip string: CuvantDeGhicit (în care se va stoca cuvântul ales aleator din fișierul cuvinte.txt) și CuvantEcran (în care se va stoca o versiune a cuvântului cu caracterele neîncălcate înlocuite cu simbolul "_" până când acestea sunt ghicite).
- srand(time(NULL));: se inițializează generatorul de numere aleatorii cu o valoare diferită la fiecare rulare a programului, bazată pe timpul curent al sistemului.
- ifstream fisier("cuvinte.txt");: se deschide fișierul cuvinte.txt pentru citire (cu ajutorul clasei ifstream).
- fisier>>numarDeCuvinte;: se citește din fișier numărul de cuvinte (acesta fiind primul element din fișier).
- numar=1+rand()%numarDeCuvinte;: se generează un număr aleatoriu între 1 și numărul de cuvinte din fișier.
- for (int i=1;i<=numar;i++) fisier>>CuvantDeGhicit;: se citește din fișier cuvântul ales (depozitat în variabila CuvantDeGhicit) pe baza numărului aleatoriu generat anterior.
- fisier.close();: se închide fișierul cuvinte.txt.
- writeln("Spanzuratoarea"); writeln("**********");: se afișează titlul jocului și o linie de separator.
- int lung=CuvantDeGhicit.length();: se calculează lungimea cuvântului ales.
- CuvantEcran=CuvantDeGhicit;: se copiază cuvântul ales în variabila CuvantEcran.
- for (i=1;i<=lung-2;i++) CuvantEcran[i]='_';: se înlocuiesc toate caracterele din CuvantEcran (cu excepția primului și ultimului caracter) cu simbolul "_" pentru a fi afișat pe ecran.
- writeln(CuvantEcran);: se afișează pe ecran CuvantEcran (cu caracterele neîncălcate înlocuite cu simbolul "_").
- unsigned int nrGreseli; char litera;: se declara doua variabile: nrGreseli (în care se va ține evidența numărului de greșeli ale utilizatorului) și litera (în care se va citi litera introdusă de utilizator).
- nrGreseli=0; bool gasita;: se inițializează numărul de greșeli cu 0 și se declară variabila booleana gasita, care va fi inițializată cu false, iar ulterior va fi setată pe true dacă litera introdusă de utilizator se regăsește în cuvântul de ghicit
- Se intră într-o buclă while, care se executa cat timp numarul de greseli este mai mic decat 10 si cuvantul afisat pe ecran este diferit de cuvantul ce trebuie ghicit.
- Se afiseaza mesajul "Dati o litera: " cu ajutorul functiei "write()" si se citeste o litera de la tastatura cu ajutorul functiei "read()".

- Se cauta litera citita in cuvantul ce trebuie ghicit cu ajutorul unui loop "for" si, daca litera este gasita, se seteaza variabila "gasita" la true si se inlocuieste "_" corespunzator cuvantului afisat pe ecran cu litera gasita.
- Daca litera este gasita, se afiseaza mesajul "Litera exista." si cuvantul afisat pe ecran cu ajutorul functiei "writeln()".
- Daca litera nu este gasita, se afiseaza mesajul "Litera nu exista.", se incrementeaza numarul de greseli si se afiseaza numarul de greseli ramase cu ajutorul functiei "writeln()".
- Daca cuvantul afisat pe ecran este egal cu cuvantul ce trebuie ghicit, se afiseaza mesajul "Ai ghicit cuvantul, deci ai castigat." cu ajutorul functiei "writeln()", apoi jocul se va termina.
- Daca s-a ajuns la numărul maxim de greșeli (10), atunci se va afișa un mesaj prin care jucătorul este avertizat că a pierdut, iar jocul se va termin.

3.2. Efecte de deplasare pe ecran a propozițiilor




```
#include <string>
#include <iostream>
#include "conio2.h"

using namespace std;

int main()
{
    string s="Vremea se va incalzi usor. Vor fi usoare averse de ploaie in zonele de
munte.";

    int i;
    for (i=1;i<=80;i++)
    {
        gotoxy(i,1); printf("*");
        gotoxy(i,5); printf("*");
    }
    for (i=1;i<=5;i++)
    {
        gotoxy(1,i); printf("*");
        gotoxy(80,i); printf("*");
    }
    int n=s.length();
    for (i=1;i<=n;i++)
        s=" "+s+" ";
    for (i=0;i<s.length()-76;i++)
    {
        gotoxy(2,3);
```

```
        cout<<s.substr(i,78);
        delay(50);
    }
    getch();
    return 0;
}
```

Apelul metodei **substr** sub forma **sir.substr(poz,lung)** generează un subșir din șirul **sir**, începând cu poziția **poz**, de lungime **lung** caractere.

3.3. Transformarea numerelor din scrisul cu cifre în scrisul cu litere

3.4. Despărțirea cuvintelor în silabe

Ne propunem să scriem o funcție care să despartă un cuvânt din limba română în silabe.
Regulile de despărțire a cuvintelor în silabe, în limba română (sursa: https://www.limba-romana.net/lectie/Reguli-de-despartire-a-cuvintelor-in-silabe/72/)

Regula 1. O consoană între două vocale trece în silaba următoare:
ma-să; pă-du-re; u-ti-li-za-re;
Această regulă se aplică şi în următoarele situaţii:
a) dacă înaintea consoanei sau după ea se află un diftong sau un triftong, la despărţirea în silabe consoana trece în silaba următoare:
pâi-ne; stro-pea-lă; cre-ioa-ne;
b) litera x este considerată ca o singură consoană şi trece în silaba următoare atunci când se află între două vocale:
a-xă; e-xa-men; e-xer-ci-ţiu;
c) atunci când între cele două vocale se află literele CE, CI, GE, GI, CHE, CHI, GHE, GHI, care redau o singură consoană, acestea trec în silaba următoare: ve-che; în-ger; ve-he; un-gher;

Regula 2. Două consoane între două vocale: la despărţirea în silabe se despart şi ele, prima consoană trece în silaba dinainte, iar a doua consoană la silaba următoare: ar-tă; ac-tiv; tic-sit; mun-te; un-ghi-e;
Excepție: Dacă în grupul de două consoane, a doua este l sau r, iar prima este b,c,d,f,g,h,p,t,v, atunci despărţirea în silabe se face înaintea întregului grup, adică cele două consoane fac parte din aceeaşi silabă:
o-braz: o-blu: co-dru: a-flu-ent; a-fri-can; a-gra-fa;
pe-hli-van; su-ple-ţe; pa-tru; co-vrig: e-vla-vi-e;

Regula 3. Trei sau mai multe consoane între două vocale, la despărţirea în silabe prima consoană se duce la silaba dinainte, iar celelalte două trec la silaba

următoare: as-tru; mon-stru; con-struc-tor;

Excepție: În cazul grupurilor de consoane lpt, mpt, nct, ncţ, ncş, ndv, rct, rtf, stm, despărţirea se face după a doua consoană din grup:
sculp-tu-ră; somp-tu-os; linc-şii; punc-taj; func-ţi-e; arc-tic; jert-fa; ast-ma-tic; sand-viş;

Regula 4. Două vocale în hiat se despart între ele, făcând parte din silabe diferite: a-er; a-le-e; po-e-zi-e;

Atenție: când o vocală este urmată de un diftong sau de un triftong, despărţirea se face înaintea diftongului sau a triftongului:
plo-uă; le-oai-că;
dif. trif.
fra-ier; su-iau; du-ios;
dif. trif. dif.
ploa-ie; stea-ua;
dif. dif. dif. dif.

☞ **Idee**

Deoarece noi vom face o prelucrare pe șiruri de caractere, vom lucra cu litere și nu cu sunete. Așadar, anumite grupuri de vocale vor putea fi interpretate atât ca vocale în hiat, cât și ca diftong (respectiv triftong). De pildă, la întâlnirea grupului de vocale ie nu putem ști dacă ele trebuie despărțite (hiat ca în sanie -> sa-ni-e) sau nu (diftong ca în ploaie->ploa-ie).

La adresa http://www.etc.tuiasi.ro/sibm/romanian_spoken_language/ro/hiat.htm se găsesc explicații și exemple despre hiat.

Deoarece, din punct de vedere al șirurilor de caractere, regula 4 poate genera probleme, așadar, dacă aplicarea ei nu ar duce la rezultate corecte, atunci ar fi bine să fie stocate întâi excepțiile, sub forma de perechi cuvânt - cuvânt despărțit în silabe. Abia după testarea excepțiilor, se vor putea aplica regulile de mai sus. Se poate începe cu regula cea mai strictă, iar la fiecare regulă cu excepțiile (de exemplu cu excepțiile de la regula 3).Îmbunătățiri posibile

O sugestie este folosirea programului pentru a despărți în silabe un text în întregime. O altă sugestie este folosirea programului pentru a despărți în silabe cuvântul cel mai din dreapta de pe fiecare rând, în cazul alinierii unui text în mod "justify" într-un editor de texte.

3.5. Elisa

Trebuie facuta o versiune si fara grafica.

```
#include <iostream>
#include <fstream>
// #include <string>
#include <string.h>
#include <windows.h>
#include <time.h>
#include <regex>
#include <winbgim.h>

using namespace std;

string linie, linii[200];
int nrLinii;

int pozitie;
string sablon,rest;

string replica[2000];
int nrReplici, poz_bara;
string propozitie;

int stanga,sus;
```

```
void citesteLiniile()
{
    ifstream fisier("replici.txt");
    nrLinii=0;
    while (getline(fisier, linie))
        if (linie!="")
        {
            nrLinii++;
            linii[nrLinii]=linie;
            //cout<<linii[nrLinii]<<"\n";
        }
    fisier.close();
}

void desparteLiniaInSablonSiReplici()
{
    pozitie=linie.find("=>");
    sablon=linie.substr(0,pozitie);
    rest=linie.erase(0,pozitie+2);
    nrReplici=0;
    while (rest!="")
    {
        nrReplici++;
        poz_bara=rest.find("|");
        if (poz_bara>0)
        {
            replica[nrReplici]=rest.substr(0,poz_bara);
            rest.erase(0,poz_bara+1);
        }
        else
        {
            replica[nrReplici]=rest;
            rest="";
        }
    }
}

void citesteInModGrafic(string &tt)
{
    char t[2000], ts[2000], t2[2]=" ";
    char tasta;
    strcpy(t,"");
    strcpy(ts,"");
    setcolor(BLUE);
    strcpy(ts,t); t2[0]='_'; strcat(ts,t2);
    outtextxy(stanga,sus,ts);
    do {
        tasta=getch();
        //PlaySound("keyboard_unu.wav",NULL,SND_ASYNC);
        setcolor(WHITE);
        strcpy(ts,t); t2[0]='_'; strcat(ts,t2);
        outtextxy(stanga,sus,ts);
        if (tasta==8 && strlen(t)>0)
            t[strlen(t)-1]=0;
```

```
        else
            if (tasta>=32)
                {
                    t2[0]=tasta;
                    strcat(t,t2);
                }
        setcolor(BLUE);
        strcpy(ts,t); t2[0]='_'; strcat(ts,t2);
        outtextxy(stanga,sus,ts);
    } while (tasta!=13);
    setcolor(WHITE);
    strcpy(ts,t); t2[0]='_'; strcat(ts,t2);
    outtextxy(stanga,sus,ts);
    setcolor(BLUE);
    outtextxy(stanga,sus,t);
    tt=string(t);
    sus+=20;
}

void afiseazaIncet(string scpp)
{
    stanga=10;
    //PlaySound("keyboard_sound.wav",NULL,SND_ASYNC);
    setcolor(RED);
    int s; char t2[2];
    const char *t;
    t=scpp.c_str();
    int x=stanga;
    for (int i=0;i<strlen(t);i++)
    {
        s=0;
        //for (int j=0;j<i;j++) { t2[0]=t[j]; s=s+textwidth(t2); }
        x=x+textwidth("M");
        if (x>650) {x=stanga; sus+=20;}
        t2[0]=t[i];
        outtextxy(x,sus,t2);
        //delay(100);
    }
    sus+=20;
    //PlaySound(NULL,NULL,SND_SYNC);
}

int main()
{
    //string linie= "*tuse*febra*=>Mergi la un doctor!|Ia paracetamol si bea
ceaiuri.|Mergi la farmacie si intreaba de medicamentele potrivite.";
    initwindow(1200,800,"Doctor");
    setbkcolor(WHITE); cleardevice();
    readimagefile("doctor.jpg",670,0,1200,800);
    settextstyle(0,0,2);
    srand(time(NULL));
    stanga=20; sus=10;
    afiseazaIncet("Buna ziua, sunt consultantul tau in domeniul sanatatii.");
    string nume;
    afiseazaIncet("Cum te numesti?");
```

```
    citesteInModGrafic(nume);
    afiseazaIncet("Ce te deranjeaza?");
    do
    {
        citesteInModGrafic(propozitie);
        citesteLiniile();
        int i;
        bool gasit=false;
        for (i=1; i<=nrLinii; i++)
        {
            linie=linii[i];
            desparteLiniaInSablonSiReplici();
            regex sablon_regex(sablon);
            if (regex_match(propozitie,sablon_regex))
            {
                gasit=true;
                int r=1+rand()%nrReplici;
                afiseazaIncet(replica[r]);
            }
        }
        if (!gasit)
            afiseazaIncet("Detaliaza.");
        if (sus>=720)
        {
            sus=10;
            cleardevice();
            readimagefile("doctor.jpg",670,0,1200,800);
        }
    }
    while (propozitie!="");
    return 0;
}
```

Replici.txt

```
.*tuse.*febra.*=>Mergi la un doctor!|Ia paracetamol si bea ceaiuri.|Mergi la
farmacie si intreaba de medicamentele potrivite.
.*nu.*bani.*=>Ia-ti un job.|Mergi la o banca si imprumuta bani.|Vorbeste cu seful
pentru o crestere de salariu
.*doctorul.*nu.*ma.*intelege.*=>Ce te face sa crezi asta?|Poti sa mergi la un alt
medic.|Consulta un tratat de medicina.
.*racit.*=>Cum se manifesta raceala?|Cine ti-a pus diagnosticul?|Poate e vorba
despre o alergie.
.*am temperatura.*=>Vrei sa spui ca ai febra?|Tusesti?
.*tuse.*=>Posibil sa fie gripa, raceala sau o alergie.|Un medic ORL ar trebui sa te
consulte.
.*antihistaminice.*=>Da, antihistaminicele sunt bune pentru alergii.|Daca ai vreo
infectie bacteriana, va trebui sa iei antibiotice.
.*antibiotice.*=>Medicul poate prescrie antibiotice.|Daca ai o viroza respiratorie,
antibioticele nu te ajuta.|Consulta un medic.
^(?=.*\b(mancarime|eruptie cutanata|umflaturi)\b).+$=>Cred ca ai urticarie. Iti
recomand sa folosesti antihistaminice, creme topice cu hidrocortizon, evitarea
alergenilor.
^(?=.*\b(dureri de cap|greata|oboseala)\b).+$=>S-ar putea sa ai o afectiune virala.
Odihneste-te mult, bea multe lichide, ia medicamente antipiretice si
antiinflamatoare.
```

^(?=.*\b(febra|tuse|dificultati respiratorii)\b).+$=>Aceste simptome sunt sugestive pentru o infectie respiratorie. Consulta medicul, stai acasa si evita contactul cu alte persoane.
^(?=.*\b(durere in gat|febra|amigdale inflamate)\b).+$=>Este posibil sa ai amigdalita. Consulta medicul, ia medicamente antiinflamatoare si antibiotice.
^(?=.*\b(durere abdominala|diaree|febra)\b).+$=>Aceste simptome sugereaza o infectie gastro-intestinala. Bea multe lichide, ia medicamente antidiareice si evita alimentele grele si grase.
^(?=.*\b(durere de spate|furnicaturi|slabiciune musculara)\b).+$=>Aceste simptome pot fi asociate cu o afectiune a coloanei vertebrale sau cu o afectiune neurologica. Consulta medicul si efectueaza investigatii suplimentare.
^(?=.*\b(durere toracica|dispnee|palpitatii)\b).+$=>Aceste simptome pot fi asociate cu o afectiune cardiaca. Consulta medicul si efectueaza un EKG si o evaluare a functiei cardiace.
^(?=.*\b(amorteala|tulburari de vedere|tulburari de echilibru)\b).+$=>Aceste simptome pot fi asociate cu o afectiune neurologica. Consulta medicul si efectueaza o evaluare neurologica si o tomografie computerizata.
^(?=.*\b(febra|transpiratii|pierdere in greutate)\b).+$=>Aceste simptome pot fi asociate cu o afectiune interna, cum ar fi o infectie sau un cancer. Consulta medicul si efectueaza analize de sange si o evaluare a functiei organelor interne.
^(?=.*\b(dureri de cap|greata|varsaturi)\b).+$=>Aceste simptome pot fi asociate cu o afectiune neurologica, cum ar fi o migrena sau o leziune cerebrala. Consulta medicul si efectueaza o evaluare neurologica si o tomografie computerizata.
^(?=.*\b(arsuri la stomac|balonare|dureri abdominale)\b).+$=>Aceste simptome pot fi asociate cu o afectiune gastrointestinala, cum ar fi o boala inflamatorie intestinala sau o ulceratie.
^(?=.*\b(senzatie de arsura la urinar|urinare frecventa|sange in urina)\b).+$=>Exista posibilitatea sa ai o infectie urinara. Este necesar sa consulti un medic pentru a primi un tratament cu antibiotice.
^(?=.*\b(dificultati de respiratie|tuse|febra)\b).+$=>Sper ca nu e pneumonie, dar consulta un medic ce iti va recomanda antibiotice, antiinflamatoare si antipiretice.

Capitolul 4. Biblioteca grafică și exemple simple

Scopul acestei lecții este familiarizarea elevilor cu elemente foarte simple de geometrie analitică, în contextul utilizării unei biblioteci grafice foarte simple. Elevii vor fi capabili să folosească primitiva grafică line(x1,y1,x2,y2), care trasează un segment de dreaptă între două puncte de coordonate (x1,y1) și (x2,y2), pentru diferite reprezentări grafice practice, cum ar fi:

- desenarea unei table de șah sau a unei table pentru un joc de masă
- desenarea unui poligon regulat și a unei stele regulate, cu n laturi / colțuri
- desenarea unor figuri definite recursiv, precum fractalii
- trasarea graficului unei funcții oarecare
- trasarea bioritmului unei persoane
- reprezentarea grafică a felului în care se formează o imagine într-o lentilă convergentă

4.1. O bibliotecă grafică elementară

Scopul nostru este de a realiza diferite exerciții de grafică, în care accentul să-l punem pe partea matematică și de algoritmică și nu pe tehnologie sau pe elemente de programare orientată-obiect (care vor fi predate la alte cursuri). De aceea, vom utiliza o interfață grafică simplă, la care noi putem adăuga propriile funcții mai complexe, create pe baza unor funcții simple predefinite. Așadar, vom folosi cea mai simplă bibliotecă grafică, numită **WinBGIm**, care extinde funcțiile din **Graphics.h**.

Scopul nostru nu este de a folosi biblioteci grafice sofisticate, pentru a face efecte de grafică tridimensională sau jocuri cu un design grafic foarte frumos.
Pentru asemenea lucruri, puteți folosi alte biblioteci grafice precum **Dislin, SDL, SFML, OpenGL, GLUT** etc. Puteți folosi și aceste biblioteci grafice, dar sunt mai complicate decât **WinBGIm**.

1. Dacă veți folosi sistemul de operare Windows, cu CodeBlocks, o versiune mai veche (16-19), împreună cu compilatorul MinGW 32-bit, atunci:
Descărcați biblioteca grafică WinBGIm de aici: **WINBGIM.ZIP**

☞ Atenție: pe web puteți găsi și alte variante ale acestei arhive, dar e posibil să fie nefuncționale.

Dupa ce descarcati și scoateți fișierele din această arhivă, veți face următoarele lucruri:

1. Extrageți fișierele **graphics.h, winmgim.h** și **libbgi.a** din arhivă.
2. Copiați fișierele **graphics.h** și **winbgim.h** în folderul **include** din folderul compilatorului MinGW, care se găsește, cel mai probabil, aici: **c:\Program Files (x86)\CodeBlocks\MinGW\include**

3. Copiati fisierul libbgi.a in folderul lib al directorului compilatorului MinGW, care se gaseste, cel mai probabil, aici:
   C:\Program Files (x86)\CodeBlocks\MinGW\lib\
4. În Code::Blocks accesați **Settings >> Compiler … >>linker settings**
5. Faceți click pe butonul **Add** în partea **Link libraries**, apoi răsfoiți și selectați fișierul **libbgi.a**
6. În partea dreaptă (adică la Other linker options) scrieți următoarele comenzi: **-lbgi -lgdi32 -lcomdlg32 -luuid -loleaut32 -lole32 -lwinmm**
7. Faceți click pe OK.

Următorul videoclip explică foarte clar pașii pe care trebuie să-i urmați.
https://youtu.be/GM0kni4jdPY

2. Dacă veți folosi sistemul de operare Windows, cu CodeBlocks, o versiune mai nouă (20), împreună cu compilatorul MinGW 64-bit, atunci:
În acest caz, problema este că biblioteca grafică WinBGIm este realizată pe 32 biți, pe când compilatorul vostru este pe 64 biți. În acest caz, va trebui să faceți astfel:
Descărcați biblioteca grafică WinBGIm de aici: **WINBGIM.ZIP**.

☞ Atenție: pe web puteți găsi și alte variante ale acestei arhive, dar e posibil să fie nefuncționale.

După ce descarcati și scoateți fișierele din această arhivă, veți face următoarele lucruri:

1. Descărcați compilatorul pe 32 de biți de la adresa următoare: https://jmeubank.github.io/tdm-gcc/
2. Veți folosi acest compilator, pentru a putea fi recunoscută biblioteca **WinBGIm**. Mergeți în CodeBlocks la **Settings->Compiler->Toolchain Executables** și scrieți calea către noul compilator: (de exemplu: **C:\TDM-GCC-32**)
3. Apoi la **Settings->Compiler->Linker Settings** scrieți în caseta din dreapta (**Other Linker Options**) următoarele: **-lbgi -lgdi32 -lcomdlg32 -luuid -loleaut32 -lole32**
4. Faceți click pe OK.

Extrageți fișierele **graphics.h, winmgim.h** și **libbgi.a** din arhivă.
Copiați fișierele **graphics.h** și **winbgim.h** în folderul **include** din compilatorul TDM-GCC-32, care se găsește, cel mai probabil, aici: **C:\TDM-GCC-32\include**
Copiati fisierul libbgi.a in folderul lib al directorului compilatorului MinGW, care se gaseste, cel mai probabil, aici: **C:\TDM-GCC-32\lib**
Următorul videoclip explică foarte clar pașii pe care trebuie să-i urmați.
CLICK AICI

3. Dacă veți folosi sistemul de operare macOS sau Linux, cu CodeBlocks, atunci:
În locul bibliotecii grafice **WinBGIm** pentru Windows, încercați o bibliotecă grafică similară, cum ar fi **SDL_bgi**, pe care o găsiți aici:SDLBGI.

Aceasta se bazează pe biblioteca grafică performantă **SDL**, dar adaptează funcțiile pentru a fi simple și compatibile cu biblioteca grafică **WinBGIm**.

4. Dacă veți folosi sistemul de operare Windows, cu Visual Studio (Visual C++), atunci:
Descărcați pachetul corespunzător de la adresa:
https://home.cs.colorado.edu/~main/bgi/visual/.
Puteți vedea și aceste videoclipuri, pentru ajutor:

1. https://www.youtube.com/watch?v=5DN0T7YfyfM
2. https://www.youtube.com/watch?v=aRAvHudcxYc

Descrierea funcțiilor grafice

- Lista funcțiilor grafice, cu exemple la fiecare: https://home.cs.colorado.edu/~main/cs1300/doc/bgi/index.html
- Alte lucruri despre biblioteca grafică: http://www.cs.colorado.edu/~main/cs1300/doc/bgi/bgi.html.

Un exemplu de program de test pentru grafică este aici:

```
#include <iostream>
#include <graphics.h>
#include <winbgim.h>

int main()
{
// int gd = DETECT, gm;
// initgraph(&gd, &gm, "");
 initwindow(800,600); int x = 400, y = 300, r;
 for (r = 25; r <= 125; r += 20)
   circle(x, y, r);
 getch(); closegraph(); return 0;
}
```

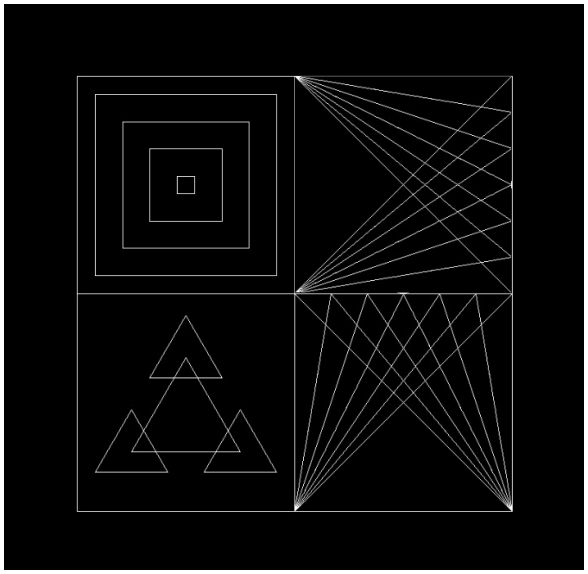
Dacă acesta funcționează, puteți continua să folosiți orice funcție grafică și de lucru cu mouse-ul din biblioteca **WinBGIm**.

4.2. Câteva exemple

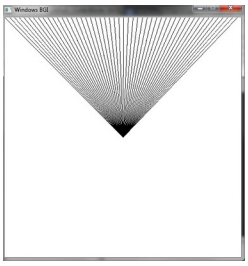
- Exemplu simplu de utilizare a graficii: click aici

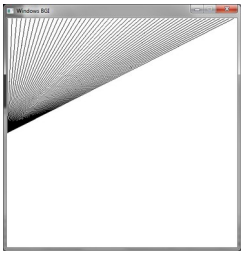
Mai jos sunt niște probleme de grafică pentru laboratorul 5, pe lângă cele date de profesorul de la laborator sau cele de pe site-ul laboratorului. De asemenea, sunt probleme de grafică și pe site-ul PRINFO.RO: https://prinfo.ro .

1. Stabiliți o fereastră de desenare cu 600 pixeli pe orizontală și 600 pe verticală, apoi împărțiți fereastra în 4 zone și în cele 4 zone realizați următorul desen.



2. Uniți prin săgeți programele din stânga cu desenele realizate de celelalte două programe, aflate în dreapta.

#include <iostream>
#include <graphics.h>
#include <winbgim.h>
int main()
{ initwindow(500,500); int i;
for (i=1;i<=250;i+=10)
line(500,500,i,1);
getch(); closegraph(); return 0;
}
	

#include <iostream>
#include <graphics.h>
#include <winbgim.h>
int main()
{ initwindow(500,500); int i;
for (i=1;i<=500;i+=10)
line(1,250,i,1);
getch(); closegraph(); return 0;
}
	

#include <iostream>
#include <graphics.h>
#include <winbgim.h>
int main()
{
initwindow(500,500); int i;
for (i=1;i<=500;i+=10) line(250,250,i,1);
getch(); closegraph();
return 0;
}	

3) Amplasarea unor cercuri cu mouse-ul

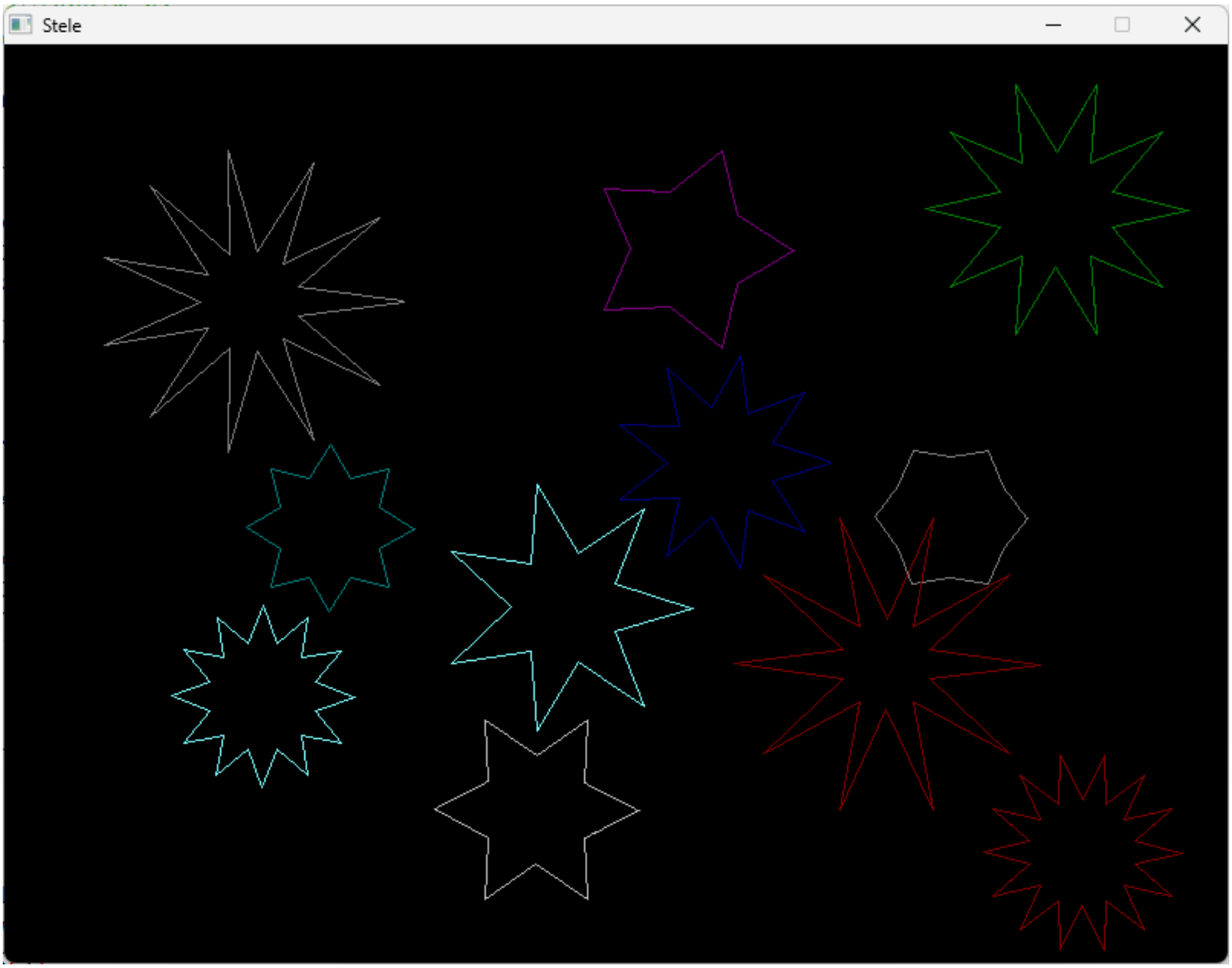
```
#include <iostream>
#include <graphics.h>
#include <winbgim.h>
```

```
#include <time.h>

using namespace std;

#define MAX 30

struct punct {
  int x, y;
};

struct cerc {
  punct centrul;
  int raza;
};

cerc Cerc[MAX];
int nrCercuri=0;

void deseneazaCerc(cerc C, int culoare)
{
  setcolor(culoare);
  circle(C.centrul.x, C.centrul.y, C.raza);
}

void amplaseazaCerc()
{
  int x,y;

    nrCercuri++;
    Cerc[nrCercuri].centrul.x=x;
    Cerc[nrCercuri].centrul.y=y;
    Cerc[nrCercuri].raza=30+rand()%41;
    deseneazaCerc(Cerc[nrCercuri],WHITE);
  }
}

int main()
{
  srand(time(NULL));
  initwindow(1000,800);
  do {
    amplaseazaCerc();
  } while (! kbhit());
  for (int pas=1; pas<=50; pas++)
  {
    for (int i=1;i<=nrCercuri;i++)
      deseneazaCerc(Cerc[i],rand()%15+1);
    delay(100);
  }
  closegraph(); return 0;
}
```

4.3. Desenarea unui poligon și a unei stele regulate

- Desenarea unui poligon regulat: click aici
- Desenarea unei stele regulate: click aici

Definiți o structură de date adecvată pentru a reprezenta o stea regulată cu un număr de colțuri (cel puțin 3), cu marginea de o anumită culoare și grosime, cu o anumită culoare în interior, cu o anumită mărime și o anumită „deschidere” a colțurilor, cu o anumită amplasare sub un anumit unghi față de orizontală a primului colț. Se pot folosi funcțiile trigonometrice sin și cos, din math.h. Scrieți o funcție care să deseneze o asemenea stea, oarecare, amplasată oriunde și oricum, de culoare oarecare etc.




```
#include <time.h>
#include <math.h>
#include <winbgim.h>
#define pi 3.14159265

using namespace std;

void stea(int x, int y, int R, int r, int n, float gama, int culoare)
{
  setcolor(culoare);
  int X[100], Y[100];
```

```
  float alfa=pi/n, beta;
  int i;
  int raza;
  for (i=0; i<=2*n; i++)
  {
    if (i % 2 == 0)
      raza=R;
    else
      raza=r;
    beta=alfa*i+gama;
    X[i]=x+raza*cos(beta);
    Y[i]=y+raza*sin(beta);
    // circle(X[i],Y[i],4);
  }
  for (i=0;i<2*n;i++)
    line(X[i],Y[i],X[i+1],Y[i+1]);
}

int main()
{
  srand(time(NULL));
  initwindow(800,600);
  int x,y;
  do
  {
    if(ismouseclick(WM_LBUTTONDOWN))
    {
      clearmouseclick(WM_LBUTTONDOWN);
      x=mousex();
      y=mousey();
      stea(x,y,50+rand()%51, 30+rand()%11, 5+rand()%10, 0, 1+rand()%15);
    }
  } while (!kbhit());
  /*
  float gama=0;
  while (!kbhit())
  {
    stea(400,300,200,100,5,gama,WHITE);
    delay(100);
    stea(400,300,200,100,5,gama,BLACK);
    gama=gama+0.05;

  }
  */
  getch();
  closegraph();
  return 0;
}
```

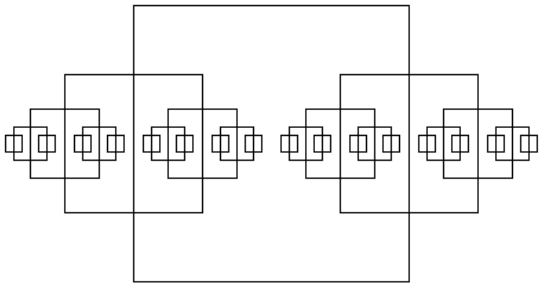
4.4. Figuri recursive cu cercuri si patrate

Probleme rezolvate

```
void patrat(int x, int y, int lat)
{
  line(x-lat/2,y-lat/2,x-lat/2,y+lat/2); line(x-lat/2,y+lat/2,x+lat/2,y+lat/2);
  line(x+lat/2,y+lat/2,x+lat/2,y-lat/2); line(x+lat/2,y-lat/2,x-lat/2,y-lat/2);
}

void diamantPatrat(int x, int y, int lat)
{
  if (lat>40)
  {
   patrat(x,y,lat);
   diamantPatrat(x-lat/2,y+lat/2,lat/2); diamantPatrat(x+lat/2,y+lat/2,lat/2);
   diamantPatrat(x+lat/2,y-lat/2,lat/2); diamantPatrat(x-lat/2,y-lat/2,lat/2);
  }
}

void diamantCerc(int x, int y, int lat)
{
  if (lat>40)
  {
   circle(x,y,lat);
   diamantCerc(x-lat/2,y,lat/2); diamantCerc(x,y+lat/2,lat/2);
   diamantCerc(x+lat/2,y,lat/2); diamantCerc(x,y-lat/2,lat/2);
  }
}
```

Probleme propuse

1) Care din următoarele funcții “diamant” desenează o figură ca cea din stânga următoare, dacă este inclusă în programul din dreapta?



a) funcția 1
b) funcția 2
c) funcția 3
d) funcția 4

```
#include <iostream>
#include <graphics.h>
#include <winbgim.h>
void patrat(int x,int y, int L)
{
  int L2=L/2;
  line(x-L2,y-L2,x+L2,y-L2);
  line(x+L2,y-L2,x+L2,y+L2);
  line(x+L2,y+L2,x-L2,y+L2);
  line(x-L2,y+L2,x-L2,y-L2);
}
int main()
{
  initwindow(800,600);
  diamant(400,300,200);
  getch(); closegraph();
  return 0;
}
```

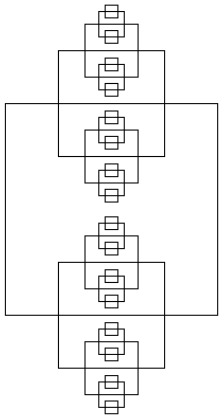
```
// functia 1
void diamant(int x, int y, int L)
{
  if (L>10) {
      patrat(x,y,L);
      diamant(x+L/2,y,L/4);
      diamant(x-L/2,y,L/4); }
}
```

```
// functia 2
void diamant(int x, int y, int L)
{
  if (L>10) {
      patrat(x,y,L);
      diamant(x-L,y,L/2);
      diamant(x+L,y,L/2); }
}
```

```
// functia 3
void diamant(int x, int y, int L)
{
  if (L>10) {
      patrat(x,y,L);
      diamant(x-L/2,y,L/2);
      diamant(x+L/2,y,L/2); }
}
```

```
// functia 4
void diamant(int x, int y, int L)
{
  if (L>10) {
      patrat(x,y,L);
      diamant(x,y+L/2,L/2);
      diamant(x,y-L/2,L/2); }
}
```

2) Care din următoarele funcții "diamant" desenează o figură ca cea din stânga următoare, dacă este inclusă în programul din dreapta?



a) funcția 1
b) funcția 2
c) funcția 3
d) funcția 4

```
#include <iostream>
#include <graphics.h>
#include <winbgim.h>
void patrat(int x,int y, int L)
{
  int L2=L/2;
  line(x-L2,y-L2,x+L2,y-L2);
  line(x+L2,y-L2,x+L2,y+L2);
  line(x+L2,y+L2,x-L2,y+L2);
  line(x-L2,y+L2,x-L2,y-L2);
}
int main()
{
  initwindow(800,600);
  diamant(400,300,200);
  getch(); closegraph();
  return 0;
}
```

```
// functia 1
void diamant(int x, int y, int L)
{
  if (L>10) {
      patrat(x,y,L);
      diamant(x-L,y,L/2);
      diamant(x+L,y,L/2); }
}
```

```
// functia 2
void diamant(int x, int y, int L)
{
  if (L>10) {
      patrat(x,y,L);
      diamant(x+L/2,y,L/4);
      diamant(x-L/2,y,L/4); }
}
```

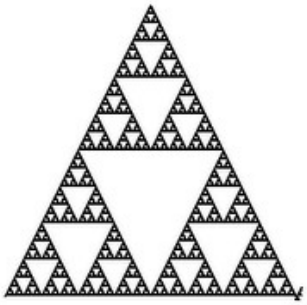
```
// functia 3
void diamant(int x, int y, int L)
{
  if (L>10) {
```

```
// functia 4
void diamant(int x, int y, int L)
{
  if (L>10) {
```

```
      patrat(x,y,L);
      diamant(x-L/2,y,L/2);
      diamant(x+L/2,y,L/2); }
}
```

```
      patrat(x,y,L);
      diamant(x,y+L/2,L/2);
      diamant(x,y-L/2,L/2); }
}
```

3) Desenați recursiv acest triunghi (numit triunghiul Sierpinski).



4.5. Trasarea graficului unei funcții oarecare

Ne propunem, în continuare, să realizăm graficul unei funcţii oarecare f:R->R (definită în cadrul programului), într-o zonă dreptunghiulară (x1,y1,x2,y2) oarecare a ecranului, cu y1>y2. Funcţia va fi reprezentată pe un anumit interval [a,b], deci x aparține lui [a,b].
Se vor stabili, mai întâi, valorile minimă şi maximă ale funcţiei, pe intervalul [a,b]. Să zicem că ele sunt ymin şi ymax. Atunci vom face o transformare a imaginii reale a graficului în imaginea din zona de pe ecran a acestui grafic.

Această transformare va duce punctul P(x,y) din planul real în punctul Q(xe,ye) din planul ecranului calculatorului. De aceea vom utiliza două transformări liniare, care vor duce pe x în xe, iar pe y în ye. Funcţiile pe care le vom folosi vor avea forma m*x+n.

Pentru a determina - pentru ambele transformări - pe m şi n, va trebui să aplicăm formulele de transformare pentru capetele intervalelor. Aşadar, având: xe=m*x+n, obţinem x1=m*a+n, respectiv x2=m*b+n, de unde se determină m şi n, deci xe în funcţie de x.
La fel se determină ye, în funcţie de y. În final, se vor obţine formulele:

- xe=(x2-x1)*x/(b-a)+(b*x1-a*x2)/(b-a)
- şi ye=(y2-y1)*y/(ymax-ymin)+(ymax*y1-ymin*y2)/(ymax-ymin).

Valorile xe şi ye vor fi rotunjite la numere întregi, deoarece în ecranul grafic se lucrează cu coordonate întregi şi nu reale.

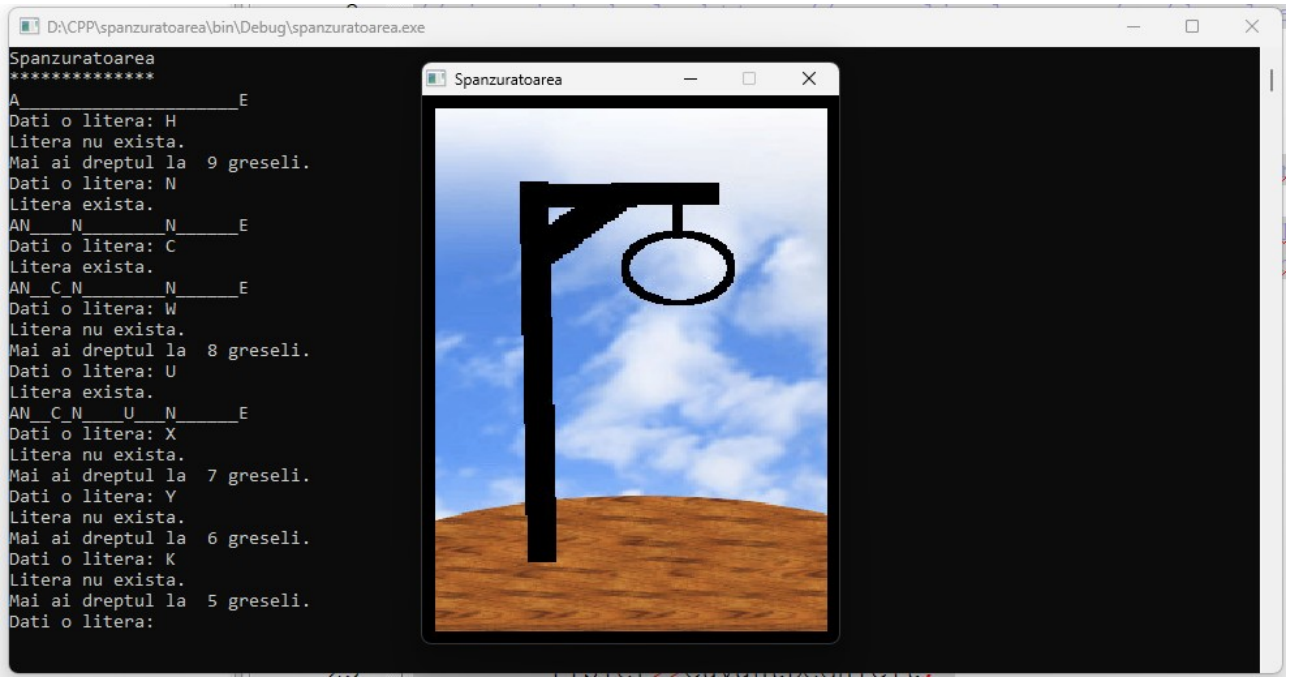
☞ **Video:** Graficul unei funcții (21:33): click aici

Propunem să combinați programul care va reprezenta grafic o funcție cu unul care să verifice corectitudinea sintactică a unei funcţii, precum şi cu un program care să evalueze expresii.

Astfel, veți putea realiza un program care să reprezinte grafic orice funcţie, a cărei expresie să fie dată de la tastatură. În loc ca funcția să fie parte componentă din codul programului, ar apărea o funcție C/C++ care să evalueze o funcție matematică într-un punct oarecare, funcția matematică fiind exprimată ca un șir de caractere.

Capitolul 5. Aplicații cu grafică

5.1. Jocul Spânzurătoarea cu grafică



Vom realiza un program C++ care implementează jocul Spânzurătoarea, folosind biblioteca grafică WinBGIm. Programul începe prin declarația unor variabile de tip string pentru cuvântul de ghicit și cuvântul afișat pe ecran, respectiv prin importul unor biblioteci necesare pentru funcționarea programului.

```
#include <fstream>
#include <time.h>
#include <winbgim.h>
#include <string>
#include <pcr.h>

using namespace std;

int main()
{
  string CuvantDeGhicit, CuvantEcran;
  ...
}
```

În continuare, programul utilizează funcția srand pentru a inițializa generatorul de numere aleatoare, iar apoi deschide fișierul "cuvinte.txt" și citeste din el numărul total de cuvinte și apoi selectează un cuvânt aleatoriu din fișier.

```
  srand(time(NULL));
  ifstream fisier("cuvinte.txt");
  int numarDeCuvinte, numar;
  fisier>>numarDeCuvinte;
  numar=1+rand()%numarDeCuvinte;
  for (int i=1;i<=numar;i++)
```

```
    fisier>>CuvantDeGhicit;
  fisier.close();
```

Se afișează pe ecran un mesaj de bun venit, iar cuvântul de ghicit este afișat sub forma de liniuțe, cu prima și ultima literă preluate din cuvântul ce trebuie ghicit.

```
  writeln("Spanzuratoarea");
  writeln("***************");
  int lung=CuvantDeGhicit.length();
  int i;
  CuvantEcran=CuvantDeGhicit;
  for (i=1;i<=lung-2;i++)
    CuvantEcran[i]='_';
  writeln(CuvantEcran);
```

Urmează inițializarea unor variabile pentru numărul de greșeli, caracterul citit de la tastatură și numele fișierului de imagine a spanzurătorii. Se deschide fereastra grafică și se afișează prima imagine.

```
  unsigned int nrGreseli;
  char litera;
  string poza;
  nrGreseli=0;

  initwindow(320,420,"Spanzuratoarea");
  readimagefile("0.jpg",10,10,310,410);
  rectangle(10,10,310,410);
```

Se deschide fișierul "cuvinte.txt" pentru citire, ce conține mai multe cuvinte. Se citeste numarul de cuvinte din fisier si se generează un număr aleatoriu între 1 și numărul de cuvinte, care va fi poziția cuvântului ce trebuie ghicit, în fișier. Se citeste din fișier cuvântul ce trebuie ghicit. Se închide fișierul "cuvinte.txt".

Pentru jocul propriu-zis se utilizează un ciclu while pentru a permite utilizatorului să ghicească cuvântul. În fiecare iterație, programul citește un caracter de la tastatură și verifică dacă acesta apare în cuvântul de ghicit. Dacă da, caracterul este afișat în cuvântul afișat pe ecran, iar dacă nu, este afișată următoarea imagine a spanzurătorii.

- Vom parcurge cuvântul ce trebuie ghicit și vom verifica dacă litera introdusă de jucător se află în cuvânt. Dacă se găsește litera în cuvânt, variabila "gasita" devine "true", iar litera respectivă este afișată pe ecran în locul '_'-urilor.
- Dacă litera introdusă de jucător se găsește în cuvânt, se afișează mesajul "Litera exista." și se afișează pe ecran cuvântul cu literele deja ghicite.
- Dacă litera introdusă de jucător nu se găsește în cuvânt, se afișează mesajul "Litera nu exista.", se incrementează numărul de greșeli ("nrGreseli") și se afișează pe ecran imaginea corespunzătoare numărului de greșeli. De

asemenea, se afișează și numărul de încercări rămase până la pierderea jocului.

Va trebui să verificăm dacă jucătorul a ghicit cuvântul sau a pierdut jocul și se afișează un mesaj corespunzător.
La final, se va afișa mesajul "Mergi in fereastra grafica si apasa o tasta." și se va așteapta ca jucătorul să apese o tastă pentru a închide fereastra grafică, apoi se va închide fereastra grafică și se va termină programul.

Codul complet al programului

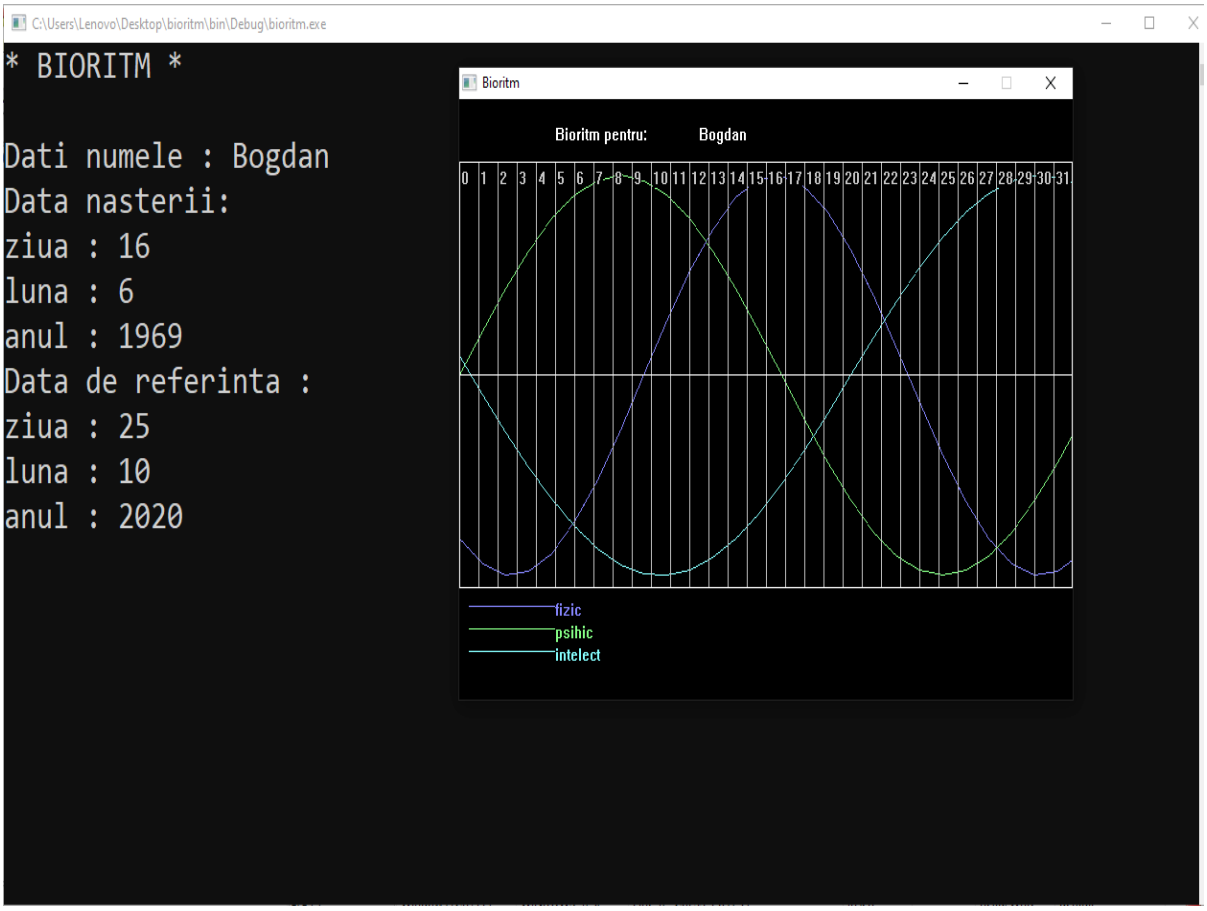
```
// Spanzuratoarea
// imagini de la https://www.oligalma.com/en/downloads/images/hangman
// 0.jpg .... 10.jpg
#include <fstream>
#include <time.h>
#include <winbgim.h>
#include <string> // folosim clasa string pentru siruri de caractere
#include <pcr.h>
// functiile read (variabila de orice fel), write(e1,e2,...,en) si
writeln(e1,e2,...,en)
// afiseaza expresii de tipuri diferite, iar writeln trece si pe randul urmator

using namespace std;

int main()
{
  string CuvantDeGhicit, CuvantEcran;
  srand(time(NULL));
  ifstream fisier("cuvinte.txt");
  int numarDeCuvinte, numar;
  fisier>>numarDeCuvinte;
  numar=1+rand()%numarDeCuvinte;
  for (int i=1;i<=numar;i++)
    fisier>>CuvantDeGhicit;
  fisier.close();
  writeln("Spanzuratoarea"); writeln("***************");
  //write("Dati cuvantul ce trebuie ghicit: ");
  // read(CuvantDeGhicit);
  int lung=CuvantDeGhicit.length();
  int i;
  CuvantEcran=CuvantDeGhicit;
  for (i=1;i<=lung-2;i++)
    CuvantEcran[i]='_';
  writeln(CuvantEcran);

  unsigned int nrGreseli;
  char litera;
  string poza;
  nrGreseli=0;
  bool gasita;

  initwindow(320,420,"Spanzuratoarea");
  readimagefile("0.jpg",10,10,310,410);
```

```
  rectangle(10,10,310,410);

  while (nrGreseli<10 && CuvantEcran!=CuvantDeGhicit)
  {
    write("Dati o litera: ");
    read(litera);

    gasita=false;
    for (i=1;i<lung-1;i++)
      if (CuvantDeGhicit[i]==litera)
      {
        gasita=true;
        CuvantEcran[i]=litera;
      }

    if (gasita)
    {
      writeln("Litera exista.");
      writeln(CuvantEcran);
    }
    else
    {
      writeln("Litera nu exista.");
      nrGreseli++;
      poza=to_string(nrGreseli)+".jpg"; // 8 -> "8" + ".jpg" -> "8.jpg"
      readimagefile(poza.c_str(),10,10,310,410);
      writeln("Mai ai dreptul la ",10-nrGreseli,"greseli.");
    }
  }

  if (CuvantEcran==CuvantDeGhicit)
    writeln("Ai ghicit cuvantul, deci ai castigat.");
  else
    writeln("Nu ai ghicit cuvantul, deci ai pierdut.");

  writeln("Mergi in fereastra grafica si apasa o tasta.");
  getch();
  closegraph();
  return 0;
}
```

5.2. Trasarea grafică a bioritmului unei persoane

Specialişti din diferite domenii ale ştiinţelor care au ca obiect de studiu omul, printre care şi medicina, au ajuns la concluzia că puterea fizică, psihică şi intelectuală a oricărei persoane evoluează funcţional sub forma unor sinusoide, ale căror perioade sunt de 23, 28 şi 33 de zile, iar începutul sinusoidelor depind de data naşterii; pentru o persoana născută pe 1 ianuarie în anul 1, se consideră că bioritmul său porneşte cu cele trei curbe de la 0 (medie), în sens crescător. Pornind de la ideile de mai sus, prezentăm în continuare un simplu program de trasare grafică a bioritmului pentru o anumită persoană. Programul cere utilizatorului să introducă data sa de naştere, numele său, precum şi data de referinţă.

Bioritmul va fi trasat pe o perioadă de 32 de zile, începând cu data de referinţă stabilită.



Funcţia NrZile determină numărul de zile scurse de la 1 ianuarie anul 1, până la o anumită dată. Astfel se poate determina numărul de zile între data de naştere a persoanei şi data de referinţă. Programul poate fi rescris în orice altă implementare de C şi pe orice alt tip de calculator, însă de fiecare dată trebuie să se ţină cont de numărul de pixeli din suprafaţa de desenare şi de sistemul de axe xOy.
În cazul nostru, dimensiunea ecranului a fost de 640×480 pixeli, iar orientarea axei Oy a fost de sus în jos, iar a axei Ox de jos în sus. Originea axelor este în colţul din stânga-sus a zonei de desenare. Deşi calculatorul nu greşeşte, e posibil ca oamenii de ştiinţă care au inventat bioritmul să fi greşit de la bun început, aşa încât ramâne la latitudinea voastră dacă veţi lua în considerare rezultatele furnizate de acest program sau nu, tratând programul ca un simplu amuzament.

```
#include <stdio.h>
#include <graphics.h>
#include <math.h>
#include <stdlib.h>
#include <dos.h>
#define pi 3.1415926

void OpenGraph()
{
      /* int gd=0, gm; initgraph(&gd,&gm,"C:\\BC\\BGI"); */
```

```
      initwindow(640,480,"Bioritm");
}

int anBisect(int y)
{
      if (y%400==0 || y%4==0 && y%100!=0)
            return 1;
      else
            return 0;
}

int NrZile(int d, int m, int y)
{
      int n;
      const int nn[12]={0,31,59,90,120,151,181,212,243,273,304,334};
      n=nn[m-1];
      n=n+365*y+y/4 + d + 1;
      n=n-y/100 + y/400;
      if (m>2 && !anBisect(y)) n--;
      return n;
}

void Inputt(char *sir, int *nr)
{
      printf("%s",sir);
      scanf("%d",nr);
}

int main()
{
      int p[3] = {23,28,33}; int q[3];
      int m1,d1,y1,m2,d2,y2,i,n,nz,se,sen;
      float ur,sr; char nume[40]; fflush(stdin);
      printf("* BIORITM *\n\n");
      printf("Dati numele : "); gets(nume);
      printf("Data nasterii:\n");
      Inputt("ziua : ",&d1); Inputt("luna : ",&m1); Inputt("anul : ",&y1);
      printf("Data de referinta :\n"); Inputt("ziua : ",&d2);
      Inputt("luna : ",&m2); Inputt("anul : ",&y2);
      nz=NrZile(d1,m1,y1);
      nz=NrZile(d2,m2,y2)-nz;
      OpenGraph();
      outtextxy(100,20,"Bioritm pentru: "); outtextxy(250,20,nume);
      for (i=0; i<3; i++)
      {
            setcolor(9+i); line(10,405+18*i,100,405+18*i);
            if (i==0) outtextxy(100,400+18*i,"fizic");
            else if (i==1) outtextxy(100,400+18*i,"psihic");
            else outtextxy(100,400+18*i,"intelect");
            q[i]=nz % p[i]; ur=2*pi*q[i]/p[i];
            sr=sin(ur); se=(sr+1)*80;
            for (n=nz; n<=nz+31; n++)
                  {
                        q[i] = n % p[i]; ur = 2*pi*q[i]/p[i];
                        sr=sin(ur); sen=(sr+1)*160;
```

```
                        line(24*(n-nz-1),480-(100+se),24*(n-nz),480-(100+sen));
                        se=sen;
                        delay(30);
                  }
      }
      setcolor(7);
      char nr_zi[2];
      for (i=0; i<=31; i++)
            {
                  itoa(i, nr_zi, 10);
                  line(20*i,50,20*i,390);
                  outtextxy(20*i+2,55,nr_zi);
                  delay(30);
            }
      line(getmaxx(),50,getmaxx(),390);
      setcolor(15);
      line(0,220,getmaxx(),220);
      rectangle(0,50,getmaxx(),390);
      getch();
      closegraph();
}
```

5.3. Ceas digital si analogic

5.4. Morphing

Suplimentar:

5.5. Raliu auto

```
#include <winbgim.h>
#include <graphics.h>

using namespace std;

#define MAX 10

int latime;

struct viteza {
    unsigned int treapta;
    unsigned int lin1, col1, lin2, col2;
    unsigned int culoare;
    // (lin1,col1) ---> (lin2,col2), treapta=3
};

viteza vit[MAX];

void deseneazaTabla(int n, int m)
{
    rectangle(1,1,getmaxx()-1,getmaxy()-1);
    for (int i=1; i<n; i++)
        for (int j=1; j<m; j++)
            rectangle(latime*j,latime*i,latime*(j+1), latime*(i+1));
}

void myLine(int x1, int y1, int x2, int y2, int culoare)
{
    setcolor(culoare);
    line(x1*latime,y1*latime,x2*latime,y2*latime);
    circle(x2*latime,y2*latime,6);
}

void initViteza(viteza & v, int i)
{
    v.treapta=1;
    v.culoare=i+8;
    v.lin1=i; v.lin2=i+1; v.col1=1; v.col2=2;
}

void deseneazaViteza(viteza v)
{
    myLine(v.lin1,v.col1,v.lin2,v.col2,v.culoare);

}

int main()
{
    int nrLinii,nrColoane;
    nrLinii=15; nrColoane=25;
```

```
    initwindow(800,600);
    int latimeOriz=getmaxx()/nrColoane;
    int latimeVert=getmaxy()/nrLinii;
    latime=min(latimeOriz,latimeVert);
//    setcolor(WHITE);
    deseneazaTabla(nrLinii,nrColoane);

    int nrViteze=5;
    for (int i=1; i<=5; i++)
        {
            initViteza(vit[i],i);
            deseneazaViteza(vit[i]);
        }
    getch();
    closegraph();
    return 0;
}
```

5.6. Formarea imaginilor în lentilele convergente

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

Ne propunem să realizăm un program didactic pentru orele de fizică (optică), care să evidenţieze formarea imaginilor în lentilele convergente, în oricare din cele două cazuri: când obiectul se află între lentilă şi distanţa focală, respectiv după distanţa focală. Modul de utilizare a programului este simplu: se fixeaza mai intâi abscisa lentilei (lentila), apoi se mişcă mouse-ul până se stabileşte înalţimea dorită a lentilei. Mouse-ul este coborât apoi forţat pe axă şi se stabileste focarul din stânga al lentilei (abscisa sa este focar = distanţa focală).

☞ În sfârşit se stabileşte şi abscisa obiectului şi înălţimea sa: (obiect, h1). Se determină apoi distanţa x2 a imaginii faţă de lentilă, folosind *legea lentilelor*: 1/f = 1/x1-1/x2,

- unde f este distanţa focală,
- x1 este distanţa obiectului faţă de lentilă,
- iar x2 este distanţa imaginii faţă de lentilă (chiar dacă aceasta este reală (negativă) sau imaginară (pozitivă).

Pe masură ce se stabileşte înălţimea lentilei (h1), programul reprezintă grafic şi imaginea formată.

```
#include <graphics.h> #include <dos.h>
#include <stdlib.h> #include "mouse.h"
void OpenGraph()
{
        int gd=0, gm; initgraph(&gd,&gm,"c:\\bc\\bgi");
}
int lentila, hl; int focar, f; int obiect, x1, x2;
int h1, h2; void StabilesteLentila()
{
        int b,x,y; do {
               MouseData(b,x,y); MouseMove(x,240);
```

```
} while (b!=1); lentila=x;
delay(500); MouseHide(); setwritemode(1); do {
       MouseData(b,x,y); MouseMove(lentila,y); hl=240-y; //
       line(lentila,240-hl,lentila,240+hl);
```

```
            MouseShow(); delay(50); MouseHide();
            line(lentila,240-hl,lentila,240+hl);
      } while (b!=1); setwritemode(0);
      line(x,240-hl,x,240+hl); line(x,240-hl,x-4,240-hl+4); line(x,240-
      hl,x+4,240-hl+4); line(x,240+hl,x-4,240+hl-4); line(x,240+hl,x+4,240+hl-
      4); MouseShow();
}
void StabilesteFocar()
{
      int b,x,y; do {
            MouseData(b,x,y); MouseMove(x,240);
      } while (b!=1);
      focar=x; f=abs(lentila-focar); focar=lentila-f; MouseHide();
      circle(focar,240,2); circle(lentila+f,240,2); MouseShow();
}
void StabilesteObiect()
{
      int b,x,y; do {
            MouseData(b,x,y); MouseMove(x,240);
      } while (b!=1);
      obiect=x; x1=abs(lentila-obiect);
      obiect=lentila-x1; delay(500); MouseHide();
      setwritemode(1);
      do {
            MouseData(b,x,y); MouseMove(obiect,y); setcolor(14);
            line(obiect,240,obiect,y); h1 = -y+240; x2 = f*x1/(x1-
            f);
            h2 = h1*x2/x1; setcolor(13); line(obiect,240-
            h1,lentila,240-h1); line(lentila,240-
            h1,lentila+x2,240+h2); line(obiect,240-
            h1,lentila+x2,240+h2); setcolor(10);
            line(lentila+x2,240+h2,lentila+x2,240); MouseShow();
            delay(50); MouseHide(); setcolor(14);
            line(obiect,240,obiect,y); setcolor(13);
            line(obiect,240-h1,lentila,240-h1); line(lentila,240-
            h1,lentila+x2,240+h2); line(obiect,240-
            h1,lentila+x2,240+h2); setcolor(10);
            line(lentila+x2,240+h2,lentila+x2,240);
      } while (b!=1);
      h1=-y+240; setwritemode(0); line(obiect,240,obiect,240-h1);
      MouseShow();
}
void DeseneazaImaginea()
{
      x2 = f*x1/(x1-f); h2 = h1*x2/x1; MouseHide();
      setcolor(13); line(obiect,240-h1,lentila,240-h1);
      line(lentila,240-h1,lentila+x2,240+h2);
```

```
      line(obiect,240-h1,lentila+x2,240+h2);
      setcolor(10); line(lentila+x2,240+h2,lentila+x2,240); MouseShow();
}
void main()
{    int b,x,y;
       OpenGraph(); MouseInit(); setcolor(11); do {
             sound(300); delay(100); nosound(); MouseHide();
             cleardevice(); setcolor(15); line(0,240,639,240); MouseShow();
             delay(500); StabilesteLentila(); delay(500); StabilesteFocar();
             delay(500); StabilesteObiect(); DeseneazaImaginea();
             sound(300); delay(100); nosound();
             do { MouseData(b,x,y); } while (b==0);
             if (b==2) { closegraph(); exit(1); }
       } while (1);
}
```

Programul foloseste fişierul “mouse.h” pentru lucrul cu mouse-ul în modul grafic. Acest fişier este descris în continuare. Observaţi asemănările şi deosebirile faţă de fişierul “mouset.h” descris într-un capitol anterior şi care se referea la utilizarea mouse-ului în modul text.

```
#include <dos.h>
void MOUSE(int &a1, int &a2, int &a3, int &a4)
{struct REGPACK regs;
 regs.r_ax = a1; regs.r_bx = a2; regs.r_cx = a3; regs.r_dx = a4; intr(0x33,
 &regs); a1 = regs.r_ax; a2 = regs.r_bx;
 a3 = regs.r_cx; a4 = regs.r_dx;
}
void MouseInit(void)
{
      int a=1,b,c,d; MOUSE(a,b,c,d);
}

void MouseHide(void)
{
      int a,b,c,d; a=2; MOUSE(a,b,c,d);
}
void MouseShow(void)
{
      int a=1,b,d,c; MOUSE(a,b,c,d);
}
void MouseData(int& but, int& x, int& y)
{
      int a=3, b, c, d; MOUSE(a,b,c,d); but=b; x=c;
      y=d;
}
void MouseMove(int x, int y)
{
      int a=4, b, c=x, d=y; MOUSE(a,b,c,d);
```

7. Jocul „Raliu auto”

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

Clasa a X-a

Capitolul 6. Aplicații pentru tablouri bi- și tridimensionale (I)

Elemente introductive:

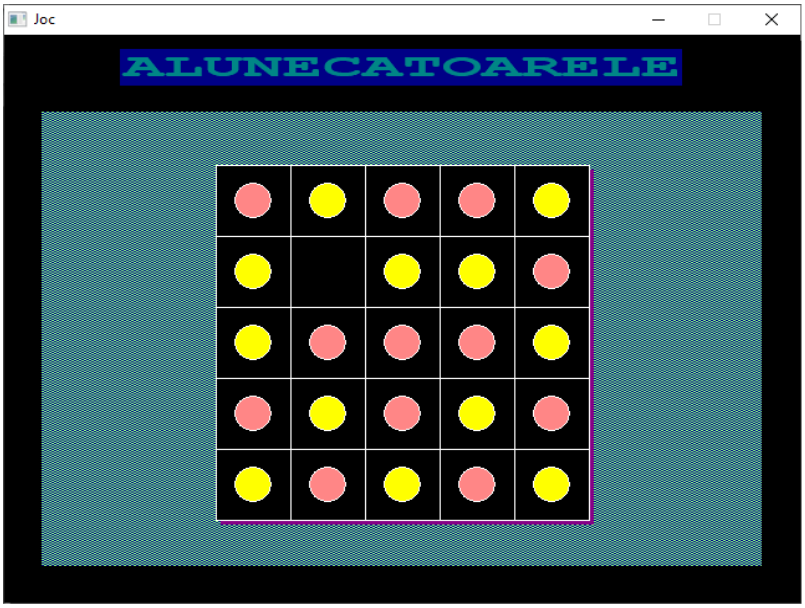
Sugestie de reprezentare a datelor pentru jocuri pe tablă (19.22)

6.1. Jocul pătratelor glisante

Un joc cu strategie câştigătoare “Jocul pătratelor glisante / alunecătoare”, pe care vi-l prezentăm în continuare, poate constitui o problemă interesantă de programare pentru orice student. Pe o tablă cu n×n pătrăţele (n întreg impar) sunt aşezate, alternativ („în șah”), piese galbene şi roşii, lăsând pătrăţelul din mijlocul tablei neocupat.
Cei doi jucători mută, alternativ, ortogonal, câte o piesă proprie alăturată spaţiului liber, în spaţiul liber.
Pierde jucătorul care nu mai poate muta (deci spaţiul liber este înconjurat numai de piese ale adversarului, eventual şi de marginea tablei).

Se poate arăta uşor că al doilea jucător are o strategie sigură de câştig, adică, dacă urmează un anumit algoritm, poate câștiga tot timpul, indiferent de ce mută primul jucător.
Pentru a explica strategia câștigătoare, să „acoperim” tabla de joc cu „dominouri” (dreptunghiuri formate din două pătrăţele elementare), lăsând căsuţa din mijloc neacoperită. Adică să „legăm” fiecare căsuță a tablei de o altă căsuță alăturată, dar fiecare căsuță aparține unei singure legături. Iar căsuța din centru să nu fie legată de nicio căsuță.
Vom observa că primul jucător, după prima mutare, va ocupa centrul şi va elibera unul din capetele unui domino. Astfel, cel de al doilea jucător va putea să mute piesa din celălalt capăt al dominoului. Deci, în orice moment, cel de al doilea jucător poate muta. Partida nu se poate termina remiză, deci primul jucător va pierde.
O metodă generală de aranjare a dominourilor, pe o tablă oarecare n×n, cu n impar, este ca dominourile să se aşeze în spirală, pornind din colţul stânga-sus şi ajungând în centrul tablei.
Această aranjare a dominourilor pe tablă va genera un vector Opus, cu semnificaţia: Opus[i]=j ⇒ dacă pătrăţelul i este liber, atunci jucătorul al doilea va muta piesa din pătrăţelul j. Pătrăţelele sunt numerotate de sus în jos, de la stânga la dreapta. Astfel, căsuţa de pe linia i şi coloana j va avea numărul (n-1)*i+j. De observat că şi Opus[j]=i.

Programul următor implementează această strategie de joc pentru calculator, în care calculatorul este întotdeauna al doilea jucător şi va câştiga întotdeauna.



O posibilă implementare a acestui joc în C/C++ poate fi cea de mai jos.

```
/* JoculPatratelorAlunecatoare */
#include <stdio.h>
/* #include "conio2.h" */
#include <graphics.h>
#include <stdlib.h>
#include <dos.h>
typedef unsigned char byte;
typedef unsigned int word;
byte Leg[26]= {0,6,7,8,5,4,1,2,3,10,9,12,11,0,15,14,21,22,19,18,25,16,17,24,23,20};
byte spatiu=0, lat=60, x0=110, y0=50;
byte is, js, Tabla[7][7];

void OpenGraph()
{
    /* int gd=DETECT,gm;
    initgraph(&gd,&gm,"c:\\bc\\bgi"); */
    initwindow(640,480,"Joc");
}

byte CasutaLibera()
{
    return (5*(is-1)+js);
}

void InitTabla()
{
    byte i,j;
    is=3;
    js=3;
    for (i=1; i<=5; i++)
        for (j=1; j<=5; j++)
            if ((i+j) % 2 != 0)
                Tabla[i][j]=LIGHTRED;
            else
                Tabla[i][j]=YELLOW;
```

```
    Tabla[is][js]=spatiu;
    for (i=0; i<=6; i++)
    {
        Tabla[0][i]=YELLOW;
        Tabla[6][i]=YELLOW;
        Tabla[i][0]=YELLOW;
        Tabla[i][6]=YELLOW;
    }
}

void DesPiesa(int i, int j)
{
    if ((i!=is) || (j!=js))
    {
        setcolor(WHITE);
        rectangle(x0+lat*i, y0+lat*j, x0+lat*i+lat, y0+lat*j+lat);
        setfillstyle(SOLID_FILL,Tabla[i][j]);
        fillellipse(x0+lat*i+lat/2, y0+lat*j+lat/2,lat/4, lat/4);
    }
    else
    {
        setfillstyle(SOLID_FILL,BLACK);
        bar(x0+lat*i, y0+lat*j, x0+lat*i+lat, y0+lat*j+lat);
        setcolor(WHITE);
        rectangle(x0+lat*i,y0+lat*j, x0+lat*i+lat, y0+lat*j+lat);
    }
}

void DesTabla()
{
    byte i,j;
    setfillstyle(SOLID_FILL,MAGENTA);
    bar(x0+lat+4,y0+lat+4,x0+lat*6+4,y0+lat*6+4);
    for (i=1; i<=5; i++)
        for (j=1; j<=5; j++)
        {
            setfillstyle(SOLID_FILL,0);
            bar(x0+lat*i,y0+lat*j,x0+lat*i+lat,y0+lat*j+lat);
            DesPiesa(i,j);
        }
}
void MutaCalc()
{
    byte isv,jsv,cas;
    isv=is;
    jsv=js;
    cas=Leg[CasutaLibera()];
    is=(cas-1)/5+1;
    js=(cas-1)%5+1;
    Tabla[is][js]=spatiu;
    DesPiesa(is,js);
    Tabla[isv][jsv]=YELLOW;
    DesPiesa(isv,jsv);
}
```

```
void MutaOm()
{
    int tasta, mut;
    do
    {
        tasta=getch();
        if (tasta==0)
            tasta=getch();
        mut=0;
        switch (tasta)
        {
        case 75:
            if (is<5)
                if (Tabla[is+1][js]==LIGHTRED)
                {
                    is++;
                    mut=1;
                    Tabla[is][js]=spatiu;
                    DesPiesa(is,js);
                    Tabla[is-1][js]=LIGHTRED;
                    DesPiesa(is-1,js);
                }
            break;
        case 77:
            if (is>1)
                if (Tabla[is-1][js]==LIGHTRED)
                {
                    is--;
                    mut=1;
                    Tabla[is][js]=spatiu;
                    DesPiesa(is,js);
                    Tabla[is+1][js]=LIGHTRED;
                    DesPiesa(is+1,js);
                }
            break;
        case 80:
            if (js>1)
                if (Tabla[is][js-1]==LIGHTRED)
                {
                    js--;
                    mut=1;
                    Tabla[is][js]=spatiu;
                    DesPiesa(is,js);
                    Tabla[is][js+1]=LIGHTRED;
                    DesPiesa(is,js+1);
                }
            break;
        case 72:
            if (js<5)
                if (Tabla[is][js+1]==LIGHTRED)
                {
                    js++;
                    mut=1;
                    Tabla[is][js]=spatiu;
                    DesPiesa(is,js);
```

```
                    Tabla[is][js-1]=LIGHTRED;
                    DesPiesa(is,js-1);
                }
            break;
        case 27:
        {
            closegraph();
            printf("Ati abandonat...");
            exit(1);
        }
        }
        /* if (!mut)
        {
        sound(200);
        delay(50);
        nosound();
        }
        */
    }
    while (!mut);
}

int EsteGata()
{
    return ((Tabla[is-1][js]==YELLOW) && (Tabla[is+1][js]==YELLOW)
            &&(Tabla[is][js-1]==YELLOW) && (Tabla[is][js+1]==YELLOW));
}

/*void Sunet()
{
 byte i;
 for (i=1; i<=25; i++)
 {
 Beep(300+20*i,30);
 delay(30);
 }
 nosound();
}
*/

int main()
{
    OpenGraph();
    setbkcolor(BLUE);
    setfillstyle(9,LIGHTGREEN);
    bar(30,65,getmaxx()-30,getmaxy()-30);
    settextstyle(DEFAULT_FONT,HORIZ_DIR,4);
    settextjustify(CENTER_TEXT,CENTER_TEXT);
    setcolor(CYAN);
    outtextxy(320,35,"ALUNECATOARELE");
    InitTabla();
    DesTabla();
    do
    {
        MutaOm();
```

```
        delay(200);
        MutaCalc();
    }
    while (!EsteGata());
    outtextxy(320,100,"AI PIERDUT.");
    /* Sunet(); */
    getch();
    closegraph();
    printf("Iarasi ati pierdut ...");
    return 0;
}
```

Pentru utilizare mouse in joc se poate folosi:

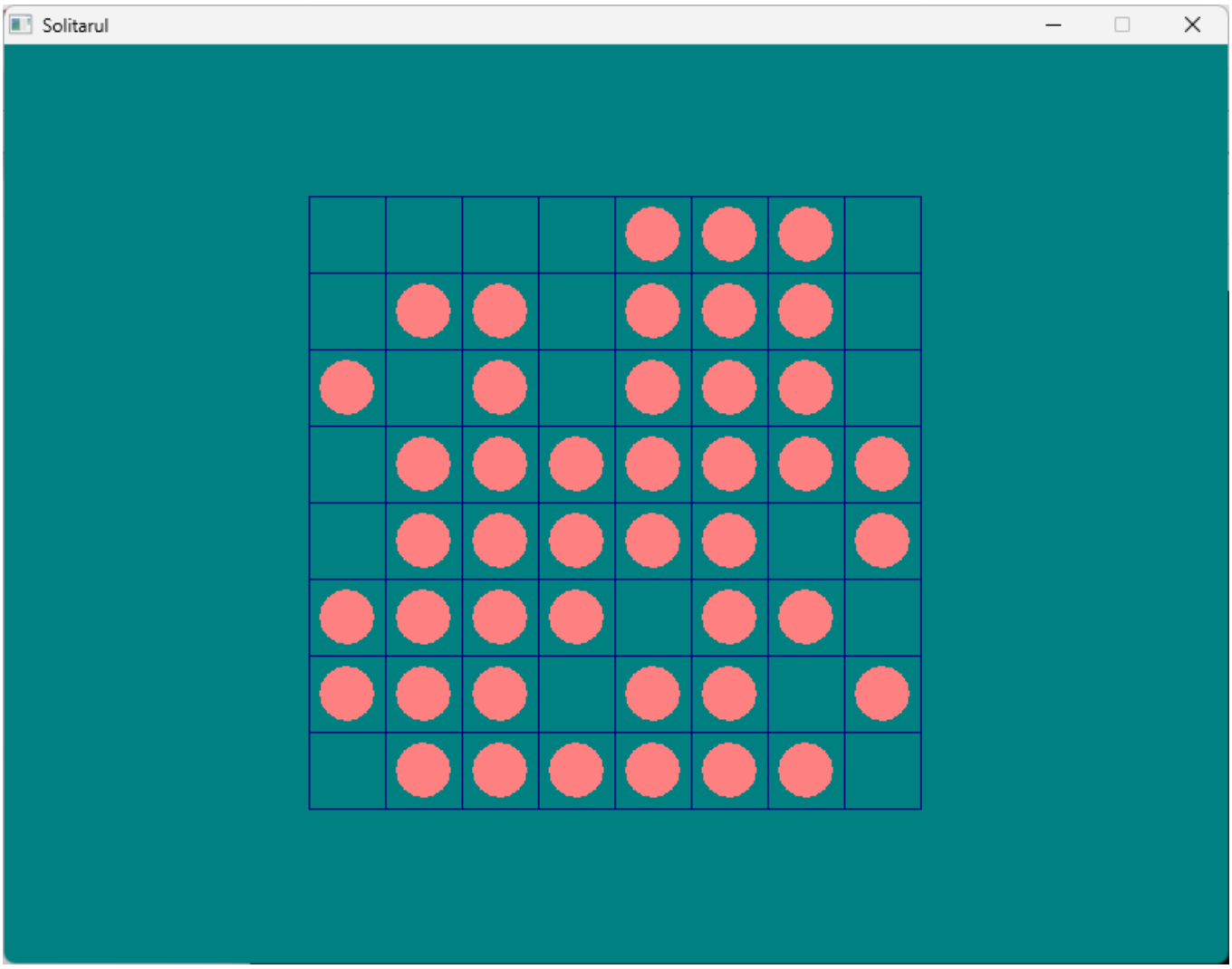
```
void utilizareMouse()
{
 int linia,coloana,x,y;
 int x1, y1, x2, y2;
 int xmijloc, ymijloc;
 if(ismouseclick(WM_LBUTTONDOWN))
 {
 clearmouseclick(WM_LBUTTONDOWN);
 x=mousex();
 y=mousey();
 if (!(x>=stanga && x<=stanga+width
  && y>=sus&&y<=sus+height))
  gata=true;
 else
 {
  linia=(y-sus)/latura+1;
  coloana=(x-stanga)/latura+1;
  if (TablaDeJoc[linia][coloana]==0)
  {
  TablaDeJoc[linia][coloana]=1;
  // stergere veche
  // desenare noua
  }
 }
 }

/* Jocul Patratelor */

#include <iostream>
#include <winbgim.h>

using namespace std;

int Leg[26]= {0,6,7,8,5,4,1,2,3,10,9,12,11,0,15,14,21,22,19,18,25,16,17,24,23,20};
int spatiu=0, lat=60, x0=110, y0=50;

int is, js, Tabla[7][7];

bool mutare;
```

```
int CasutaLibera()
{
  return (5*(is-1)+js);
}

void InitializeazaTabla()
{
  int i,j;
  is=3; js=3;
  for (i=1; i<=5; i++)
    for (j=1; j<=5; j++)
      if ((i+j) % 2 != 0)
        Tabla[i][j]=LIGHTRED;
      else
        Tabla[i][j]=YELLOW;
  Tabla[is][js]=spatiu;
  for (i=0; i<=6; i++)
  {
    Tabla[0][i]=YELLOW;
    Tabla[6][i]=YELLOW;
    Tabla[i][0]=YELLOW;
    Tabla[i][6]=YELLOW;
  }
}

void DeseneazaPiesa(int i, int j)
{
  if ((i!=is) || (j!=js))
  {
    setcolor(WHITE);
    rectangle(x0+lat*j, y0+lat*i, x0+lat*j+lat, y0+lat*i+lat);
    setfillstyle(SOLID_FILL,Tabla[i][j]);
    fillellipse(x0+lat*j+lat/2, y0+lat*i+lat/2,lat/4, lat/4);
  }
  else
  {
    setfillstyle(SOLID_FILL,BLACK);
    bar(x0+lat*j, y0+lat*i, x0+lat*j+lat, y0+lat*i+lat);
    setcolor(WHITE);
    rectangle(x0+lat*j,y0+lat*i, x0+lat*j+lat, y0+lat*i+lat);
  }
}

void DeseneazaTabla()
{
  settextstyle(DEFAULT_FONT,HORIZ_DIR,4);
  settextjustify(CENTER_TEXT,CENTER_TEXT);
  outtextxy(320,35,"JOCUL PATRATELOR");

  int i,j;
  for (i=1; i<=5; i++)
    for (j=1; j<=5; j++)
      DeseneazaPiesa(i,j);
}
```

```
void MutaCalc()
{
  int isv,jsv,cas;
  isv=is;
  jsv=js;
  cas=Leg[CasutaLibera()];
  is=(cas-1)/5+1;
  js=(cas-1)%5+1;
  Tabla[is][js]=spatiu;
  DeseneazaPiesa(is,js);
  Tabla[isv][jsv]=YELLOW;
  DeseneazaPiesa(isv,jsv);
}

void Sus()
{
  if (is<5 && Tabla[is+1][js]==LIGHTRED)
    {
      is++;
      mutare=true;
      Tabla[is][js]=spatiu;
      DeseneazaPiesa(is,js);
      Tabla[is-1][js]=LIGHTRED;
      DeseneazaPiesa(is-1,js);
    }
}

void Jos()
{
  if (is>1 && Tabla[is-1][js]==LIGHTRED)
    {
      is--;
      mutare=true;
      Tabla[is][js]=spatiu;
      DeseneazaPiesa(is,js);
      Tabla[is+1][js]=LIGHTRED;
      DeseneazaPiesa(is+1,js);
    }
}

void Dreapta()
{
  if (js>1 && Tabla[is][js-1]==LIGHTRED)
    {
      js--;
      mutare=true;
      Tabla[is][js]=spatiu;
      DeseneazaPiesa(is,js);
      Tabla[is][js+1]=LIGHTRED;
      DeseneazaPiesa(is,js+1);
    }
}

void Stanga()
{
```

```
  if (js<5 && Tabla[is][js+1]==LIGHTRED)
    {
      js++;
      mutare=true;
      Tabla[is][js]=spatiu;
      DeseneazaPiesa(is,js);
      Tabla[is][js-1]=LIGHTRED;
      DeseneazaPiesa(is,js-1);
    }
}

void MutaOm()
{
  int tasta;
  do
  {
    tasta=getch();
    if (tasta==0) tasta=getch();
    mutare=false;
    switch (tasta) {
      case 75: Stanga(); break;
      case 77: Dreapta(); break;
      case 72: Sus(); break;
      case 80: Jos(); break;
    }
  }
  while (!mutare);
}

int EsteGata()
{
  return ((Tabla[is-1][js]==YELLOW) && (Tabla[is+1][js]==YELLOW)
      &&(Tabla[is][js-1]==YELLOW) && (Tabla[is][js+1]==YELLOW));
}

int main()
{
  initwindow(640,480,"Jocul patratelor");
  InitializeazaTabla();
  DeseneazaTabla();
  do
  {
    MutaOm();
    delay(1000);
    MutaCalc();
  }
  while (!EsteGata());
  outtextxy(320,100,"AI PIERDUT.");
  getch();
  closegraph();
  return 0;
}
```

6.2. Solitarul

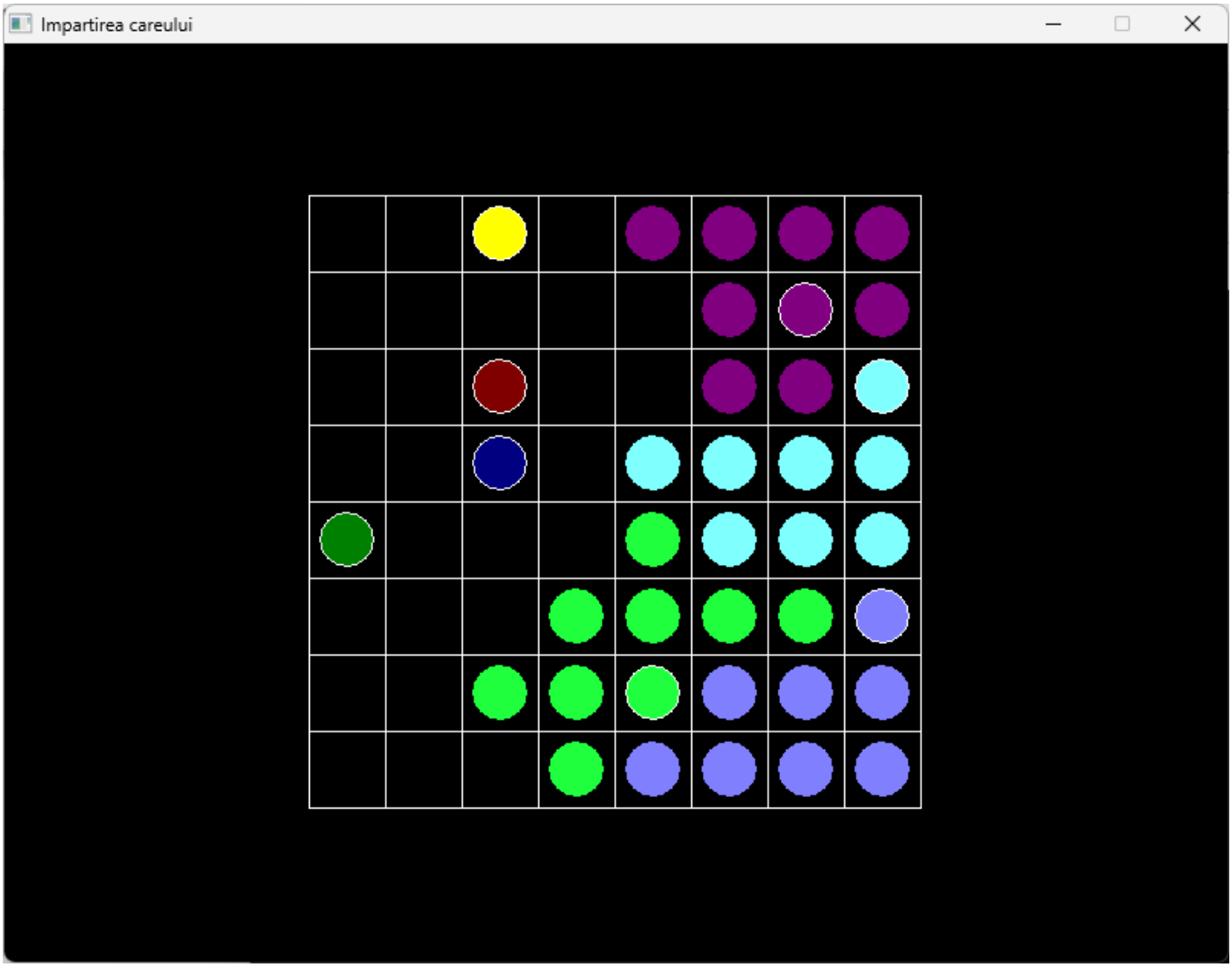



```
#include <iostream>
#include <winbgim.h>
#include <stdlib.h>

using namespace std;

#define MAX 20
#define FUNDAL CYAN

int stanga,sus,width,height,latura, numar;
bool gata;

int culoare[3]={FUNDAL,LIGHTRED,YELLOW};

int TablaDeJoc[MAX][MAX];

int scor;

void afiseazaScor()
{
    char sirScor[10];
    setcolor(FUNDAL); itoa(scor,sirScor,10);
```

```
    outtextxy(200,50,sirScor);
    int i,j; scor=0;
    for (i=1;i<=numar;i++)
        for (j=1;j<=numar;j++)
            if (TablaDeJoc[i][j]==2) scor++;
    setcolor(BLUE); itoa(scor,sirScor,10);
    outtextxy(200,50,sirScor);
}

void deseneazaPiesa(int linia, int coloana, int culoare)
{
    int x1, y1, x2, y2;
    int xmijloc, ymijloc;
    // stergere
    setcolor(FUNDAL); setfillstyle(SOLID_FILL,FUNDAL);
    x1=stanga+latura*(coloana-1); y1=sus+latura*(linia-1);
    x2=x1+latura; y2=y1+latura;
    xmijloc=(x1+x2)/2; ymijloc=(y1+y2)/2;
    bar(xmijloc-20,ymijloc-20,xmijloc+20,ymijloc+20);
    // afisez cerc rosu aprins
    setcolor(culoare); setfillstyle(SOLID_FILL,culoare);
    fillellipse(xmijloc,ymijloc,18,18);
}

void mutarePiesa(int jucator)
{
    int linia1,coloana1,linia2,coloana2,x,y;
    int x1, y1, x2, y2;
    int xmijloc, ymijloc;
    int click=false;
    do
        if(ismouseclick(WM_LBUTTONDOWN) && !click)
        {
            click=true;
            clearmouseclick(WM_LBUTTONDOWN);
            x=mousex(); y=mousey();
            linia1=(y-sus)/latura+1; coloana1=(x-stanga)/latura+1;
        }
    while (!click);
    click=false;
    do
        if(ismouseclick(WM_LBUTTONDOWN) && !click)
        {
            click=true;
            clearmouseclick(WM_LBUTTONDOWN);
            x=mousex(); y=mousey();
            linia2=(y-sus)/latura+1; coloana2=(x-stanga)/latura+1;
        }
    while (!click);
    TablaDeJoc[linia1][coloana1]=0;
    TablaDeJoc[linia2][coloana2]=jucator;
    if (linia1==linia2)
        {
            TablaDeJoc[linia1][(coloana1+coloana2)/2]=0;
            deseneazaPiesa(linia1,(coloana1+coloana2)/2,FUNDAL);
```

```
        }
    else
        {
            TablaDeJoc[(linia1+linia2)/2][coloana1]=0;
            deseneazaPiesa((linia1+linia2)/2,coloana1,FUNDAL);
        }
    deseneazaPiesa(linia1,coloana1,FUNDAL);
    deseneazaPiesa(linia2,coloana2,culoare[jucator]);
 }

bool castigat(int jucator)
{
    return false;
}

void desen()
{
   int i,j; numar=8;
   width=400; height=400; latura=width/numar;
   sus=(getmaxy()-width)/2; stanga=(getmaxx()-height)/2;
   setbkcolor(FUNDAL); clearviewport(); setcolor(BLUE);
   for (i=1; i<=numar; i++)
       for (j=1; j<=numar; j++)
           rectangle(stanga+latura*(i-1),sus+latura*(j-
1),stanga+latura*i,sus+latura*j);
}

int main()
{
   initwindow(800,600);  scor=0; desen(); afiseazaScor();
   gata=false;
   for (int i=1; i<=numar; i++)
    for (int j=1; j<=numar; j++)
        {
            if (i+j>5)
                TablaDeJoc[i][j]=1;
            else
                TablaDeJoc[i][j]=0;
            deseneazaPiesa(i,j,culoare[i+j>5]);
        }
   do
   {
      mutarePiesa(1);
      // mutarePiesa(2);
   }
   while (!castigat(1) && !castigat(2));
   getch(); closegraph();
   return 0;
}
```

6.3. Împărțirea careului

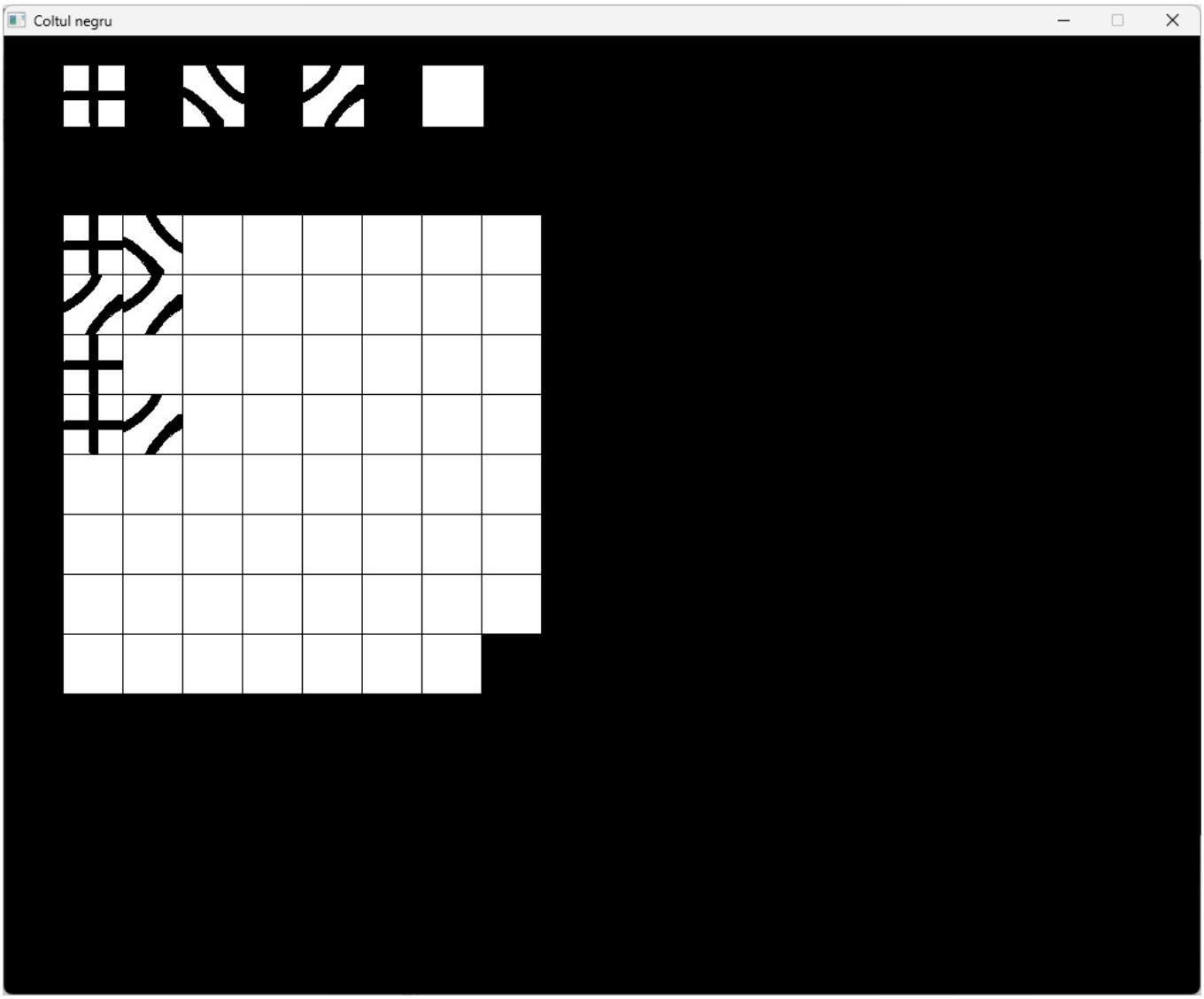



```
#include <winbgim.h>
#define MAX 20
#define FUNDAL BLACK

int latura,numar;
int width=400;
int height=400;
int stanga=100;
int sus=100;

bool gata;

int TablaDeJoc[MAX][MAX];

int culoare[8]={BLACK, RED,BLUE,GREEN, YELLOW, LIGHTCYAN, LIGHTBLUE, MAGENTA};
int culo=0;

void desenarePiesa(int linia, int coloana, int v)
{
    int c;
    if (v<0) c=-v; else c=v;
```

```
    int x1, y1, x2, y2, xmijloc, ymijloc;
    // stergere
    setcolor(FUNDAL); setfillstyle(SOLID_FILL,FUNDAL);
    x1=stanga+latura*(coloana-1); y1=sus+latura*(linia-1);
    x2=x1+latura; y2=y1+latura;
    xmijloc=(x1+x2)/2; ymijloc=(y1+y2)/2;
    bar(xmijloc-20,ymijloc-20,xmijloc+20,ymijloc+20);
    // afisez cerc de culoarea culoare[culo]
    setcolor(culoare[c]); setfillstyle(SOLID_FILL,culoare[c]);
    if (v<0) setcolor(WHITE);
    fillellipse(xmijloc,ymijloc,18,18);
}

void punerePiesa()
{
    int linia,coloana,x,y;
    if(ismouseclick(WM_LBUTTONDOWN))
    {
       clearmouseclick(WM_LBUTTONDOWN);
       x=mousex(); y=mousey();
       if (!(x>=stanga && x<=stanga+width && y>=sus && y<=sus+height))
          gata=true;
       else
       {
          linia=(y-sus)/latura+1; coloana=(x-stanga)/latura+1;
          if (TablaDeJoc[linia][coloana]>=0)
            // ==0 inseamna spatiu liber, >0 inseamna ca am o piesa de o culoare,
            // eventual de alta culoare decat cea selectata acum
             {
                TablaDeJoc[linia][coloana]=culo;
                desenarePiesa(linia,coloana,culo);
            }
          else
          if (TablaDeJoc[linia][coloana]<0)
             { // preiau culoarea de la o piesa amplasata initial
                 culo=-TablaDeJoc[linia][coloana];
             }
       }
     }
 }

void initializareTabla()
{
    numar=8;
    TablaDeJoc[3][3]=-1;
    TablaDeJoc[4][3]=-2;
    TablaDeJoc[5][1]=-3;
    TablaDeJoc[1][3]=-4;
    TablaDeJoc[3][8]=-5;
    TablaDeJoc[6][8]=-6;
    TablaDeJoc[2][7]=-7;
    TablaDeJoc[7][5]=-8;
}

void desenareTabla()
```

```
{
   int i,j;
   latura=width/numar;
   sus=(getmaxy()-width)/2; stanga=(getmaxx()-height)/2;
   setbkcolor(FUNDAL); clearviewport(); setcolor(BLUE);
   for (i=1; i<=numar; i++)
       for (j=1; j<=numar; j++)
       {
           setcolor(WHITE);
           rectangle(stanga+latura*(i-1),sus+latura*(j-
1),stanga+latura*i,sus+latura*j);
           desenarePiesa(i,j,TablaDeJoc[i][j]);
       }
}

int main()
{
   initwindow(800,600,"Impartirea careului");
   initializareTabla();
   desenareTabla();
   gata=false;
   do
   {
      punerePiesa();
   }
   while (!gata);
   getch(); closegraph();
   return 0;
}
```

6.4. Colțul negru



0.jpg

1.jpg

2.jpg

3.jpg

```
// #include <iostream>
#include <winbgim.h>
#include <string.h>

#define MAX 20

using namespace std;

int TablaDeJoc[MAX][MAX];
// 1 +  2 \\ 3 //
int n;
int piesaAleasa;
```

```
void initTabla()
{
    n=8;
    int i,j;
    for (i=1; i<=n; i++)
        for (j=1; j<=n; j++)
            TablaDeJoc[i][j]=0;
    TablaDeJoc[1][1]=1;
}

void desPiesa(int p, int l, int c)
{
    char numeFisier[10];
    switch (p)
    {
    case 0:
        strcpy(numeFisier,"0.jpg");
        break;
    case 1:
        strcpy(numeFisier,"1.jpg");
        break;
    case 2:
        strcpy(numeFisier,"2.jpg");
        break;
    case 3:
        strcpy(numeFisier,"3.jpg");
        break;
    }
    readimagefile(numeFisier, 50*c, 100+50*l, 50*(c+1)-2,100+50*(l+1)-2);
}

void desTabla()
{
    int i,j,p;
    for (i=1; i<=n; i++)
    {
        for (j=1; j<=n; j++)
        {
            p=TablaDeJoc[i][j];
            if (i!=n || j!=n) desPiesa(p,i,j);
        }
    }
}

void alegePiesa(int &p)
{
    int x,y;
    p=0;
    if(ismouseclick(WM_LBUTTONDOWN))
    {
        clearmouseclick(WM_LBUTTONDOWN);
        x=mousex();
        y=mousey();
        if (x>=50 && x<=100 && y>=25 && y<=75) p=1;
```

```
        if (x>=150 && x<=200 && y>=25 && y<=75) p=2;
        if (x>=250 && x<=300 && y>=25 && y<=75) p=3;
        if (x>=350 && x<=400 && y>=25 && y<=75) p=4;
    }
}

bool punerePiesa()
{
    int linia,coloana,x,y;
    int stanga=50, sus=150;
    int dreapta=50+50*n, jos=150+50*n;
    bool ok=false;
    if(ismouseclick(WM_LBUTTONDOWN))
    {
        clearmouseclick(WM_LBUTTONDOWN);
        x=mousex();
        y=mousey();
        if (x>=stanga && x<=dreapta && y>=sus&&y<=jos)
        {
            linia=(y-sus)/50+1;
            coloana=(x-stanga)/50+1;
            if (TablaDeJoc[linia][coloana]==0)
            {
                TablaDeJoc[linia][coloana]=piesaAleasa;
                desPiesa(piesaAleasa,linia,coloana);
                ok=true;
            }
        }
    }
    return ok;
}

void afiseazaMeniul()
{
    readimagefile("1.jpg",50,25,100,75);
    readimagefile("2.jpg",150,25,200,75);
    readimagefile("3.jpg",250,25,300,75);
    readimagefile("0.jpg",350,25,400,75);
}

int main()
{
    initwindow(1000,800,"Coltul negru");
    afiseazaMeniul();
    initTabla();
    desTabla();
    bool ok;
    do
    {
        piesaAleasa=0;
        do
        {
            alegePiesa(piesaAleasa);
            if (piesaAleasa==4)
            {
```

```
                closegraph();
                return 0;
            }
        } while (piesaAleasa==0);
        do
            ok=punerePiesa();
        while (!ok);
    }
    while (1);
    return 0;
}
```

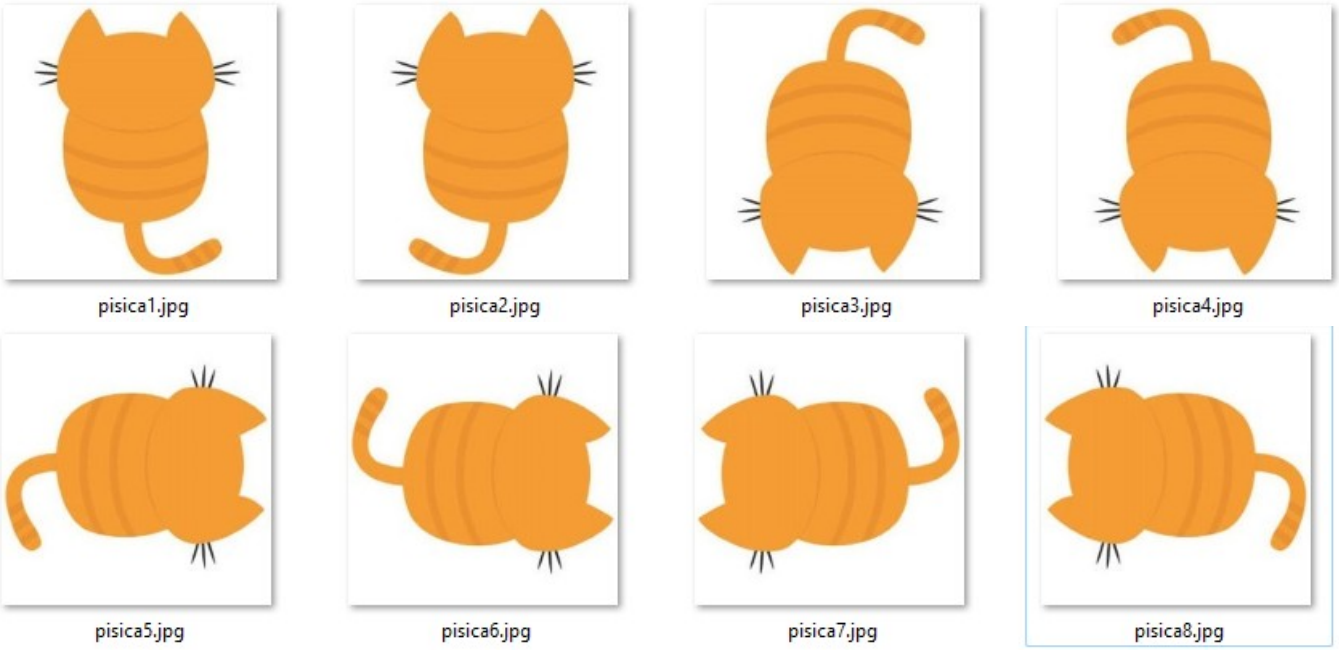
6.5. Tom și Jerry

1. Creați un fișier **„labirint.txt”**, pe care îl salvați în același director cu proiectul vostru.
Scrieți în acest fișier, separate de spații:
- numărul de linii (care este egal cu numărul de coloane)
- linia și coloana unde va fi pisica și linia și coloana unde va fi șoarecele
- apoi o matrice de caractere 1 și 0 (1 pentru zid, 0 pentru spațiu)

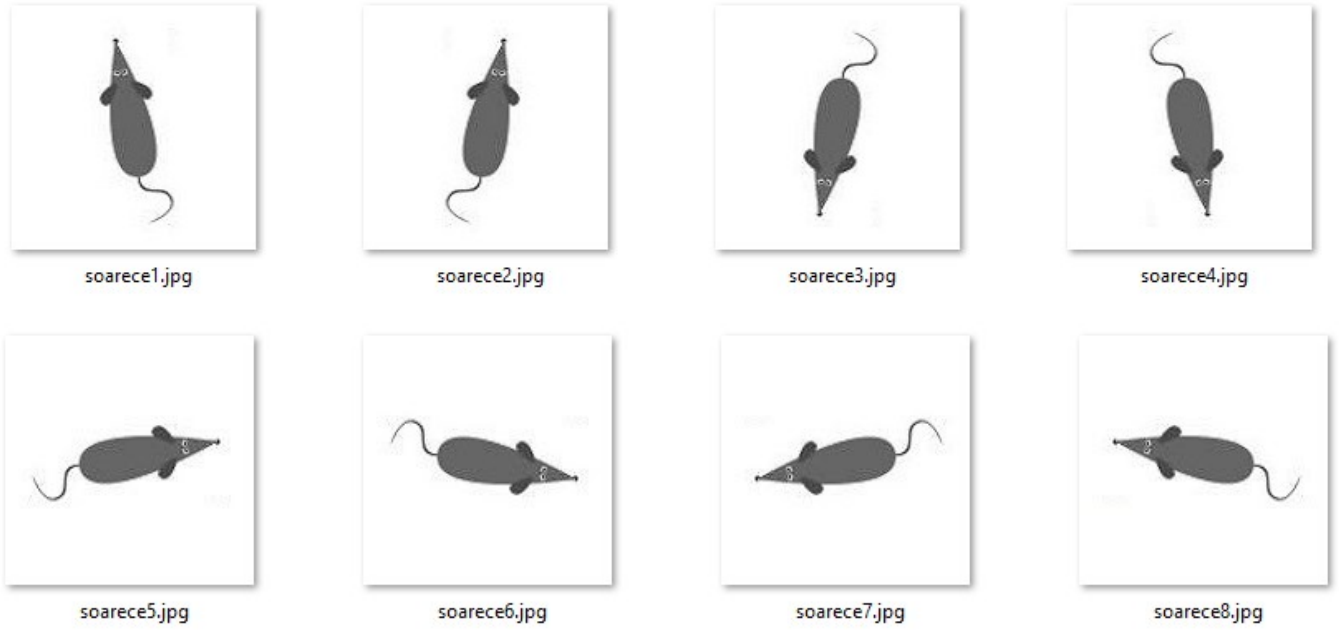
15 2 3 7 5 111111111111111 101101000001001 100001010001001 101111010000011 100010000100101 111000110101101 100010000100001 100000010000111 100100000110001 100000000001001 100011111101001 100001000101001 100011100100001 100000000000001 111111111111111	 brick.jpg

2. Copiați fișierele cu imagini ale pisicii și ale șoarecelui în diferite poziții, în directorul unde se află și proiectul vostru.

Imagini pentru pisică:



Imagini pentru șoricel:



3. Copiați fișierele cu sunetete și fișierul cu poza de început în același director.

4. Includeți în proiect bibliotecile necesare:
   - winbgim.h pentru grafică
   - fstream pentru lucrul cu fișiere
   - windows.h pentru funcția PlaySound
   - time.h pentru apelul srand(time(NULL)); care stabilește „sămânța” pentru generarea numerelor aleatorii cu ajutorul funcției rand().

5. Definiți simbolurile constante pentru tastele de săgeți și pentru tasta Escape.
SUS 72
JOS 80
STG 75
DRP 77
ESC 27
6. Definiți simbolurile constante pentru diferite direcții de deplasare, de exemplu:
DIR_STG 0
DIR_DRP 1
DIR_SUS 2
DIR_JOS 3

7. Declarați variabilele necesare:
- pentru lățimea unui pătrățel în care va apărea un zid, spațiu sau pisica/șoarecele, de exemplu:
int latime=50;
- pentru ipostaza curentă a pisicii, ipostaza curentă a șoarecelui, direcția în care merge șoarecele, direcția în care merge pisica, tasta apăsată
8. Declarați o matrice **lab**, cu cel mult 20 de linii și 20 de coloane pentru a stoca labirintul citit din fișier.
9. Scrieți o funcție void deseneaza(char car, int i, int j) care să deseneze pe linia i și coloana j poza corespunzătoare caracterului car:

- pentru ‘1’ se va citi imaginea din fișierul: brick.jpg
- pentru ‘0’ se va citi imaginea din fișierul: blank.jpg

Pentru a citit o imagine dintr-un fișier și a o afișa într-o anumită poziție se folosește funcția **readimagefile(fisier,x1,y1,x2,y2).**

10. Scrieți o funcție void deseneazaPisica() care să deseneze pisica în poziția sa curentă și în ipostaza sa curentă. Pisica se poate afla în 8 ipostaze, conform imaginilor pisica1.jpg...pisica8.jpg.
Poziția este dată de linia și coloana declarate la punctul 7.
11. Scrieți o funcție similară void deseneazaSoarece() pentru a desena șoarecele în ipostaza sa curentă, în poziția sa curentă. Se pot folosi cele 8 imagini din fișierele: soarece1.jpg...soarece8.jpg.

12. Scrieți o funcție **void stergeSoarece()** care să șteargă șoarecele din poziția sa curentă. Ștergerea șoarecelui presupune să se pună în poziția curentă a șoarecelui spațiu sau zid, în funcție de ce este în matricea labirintului în acea poziție. De exemplu, se poate apela funcția desenează astfel:
deseneaza(lab[isoarece][jsoarece],isoarece,jsoarece);
Aici matricea labirintului este lab, iar poziția șoarecelui este dată de coloana jsoarece și linia isoarece.
13. Scrieți o funcție similară **void stergePisica()** care să șteargă pisica din poziția sa curentă.
14. Scrieți funcția **void citesteLabirintul()**. Aceasta va prelua din fișierul de intrare „labirint.txt” numărul n (dimensiunea labirintului este nxn), apoi linia inițială a pisicii, coloana inițială a pisicii, linia inițială a șoarecelui, coloana inițială a șoarecelui, iar apoi toate elementele matricei **lab** (folosind două cicluri **for** de la 0 la n-1). La final închideți fișierul folosind **fisier.close()**.
15. Scrieți funcția care desenează labirintul pe baza matricei citite la punctul 14: void deseneazaLabirintul()
Se va apela în două bucle **for** funcția **deseneaza(...)** .
16. Urmează să realizați funcția **void stabilesteDirectiaSoarece()**.
Se generează un număr aleatoriu între 0 și 3, folosind funcția rand().
În funcție de acest număr, care reprezintă direcția șoarecelui, se verifică:
- dacă direcția este DIR_STG, iar șoarecele poate merge la stânga (adică nu este zid acolo), atunci se scade coloana șoarecelui cu o unitate: jsoarece--
- dacă direcția este DIR_DRP, iar șoarecele poate merge la dreapta (adică nu este zid acolo), atunci se crește coloana șoarecelui cu o unitate: jsoarece++
- la fel se procedează și cu direcțiile sus și jos.

Folosiți instrucțiunea **switch**.
17. Funcția **void stabilesteDirectiaPisica()** va stabili direcția de deplasare a pisicii în funcție de tasta apăsată.
Analizați și încercați să înțelegeți acest cod, adaptându-l după caz.

```
if (kbhit())
{
      tasta=getch();
      if (tasta==ESC) { closegraph(); exit(1); }
      if (tasta==0)
      {
```

```
            tasta=getch();
            switch(tasta) {
                  case STG: directie_pisica=DIR_STG; break;
                  case DRP: directie_pisica=DIR_DRP; break;
                  case SUS: directie_pisica=DIR_SUS; break;
                  case JOS: directie_pisica=DIR_JOS; break;
            }
            switch(directie_pisica) {
                  case DIR_STG:
                        if (jpisica>0 && lab[ipisica][jpisica-1]=='0') jpisica--;
                  break;
                  case DIR_DRP:
                        if (jpisica<n-1 && lab[ipisica][jpisica+1]=='0') jpisica++;
                        break;
                  case DIR_SUS:
                        if (ipisica>0 && lab[ipisica-1][jpisica]=='0') ipisica--;
                  break;
                  case DIR_JOS:
                        if (ipisica<n-1 && lab[ipisica+1][jpisica]=='0') ipisica++;
                  break;
            }
      }
}
```

18. Scrieți funcțiile **void schimbaIpostazaSoarece()** și **void schimbaIpostazaPisica()** ca să se adapteze direcției de deplasare a șoarecelui, respectiv a pisicii.

19. Implementați acest program în funcția **int main()**.

```
int main()
{
      srand(time(NULL));
      citesteLabirintul();
      initwindow(latime*n,latime*n,"Tom si Jerry in labirint",300,200);
      PlaySound("tomandjerry.wav",NULL,SND_ASYNC | SND_LOOP);
      readimagefile("tomjerry.jpg",0,0,latime*n,latime*n);
      getch();
      deseneazaLabirintul();
      directie_pisica=DIR_STG;
      deseneazaPisica();
      ipostaza_soarece=1;
      deseneazaSoarece();
      do {
            stergeSoarece();
            stabilesteDirectiaSoarece();
            for (int i=1;i<=2;i++)
            {
                  schimbaIpostazaSoarece();
                  deseneazaSoarece();
            delay(10);
            }
            stergePisica(); stabilesteDirectiaPisica();
            schimbaIpostazaPisica(); deseneazaPisica();
            if (ipisica==isoarece && jpisica==jsoarece)
            {
                  PlaySound("tomlaughing.wav",NULL,SND_SYNC);
```

```
                delay(5000); closegraph(); exit(0);
            }
            delay(50);
      } while (tasta!=ESC);
      getch(); closegraph();
      return 0;
}
```

Codul complet al programului

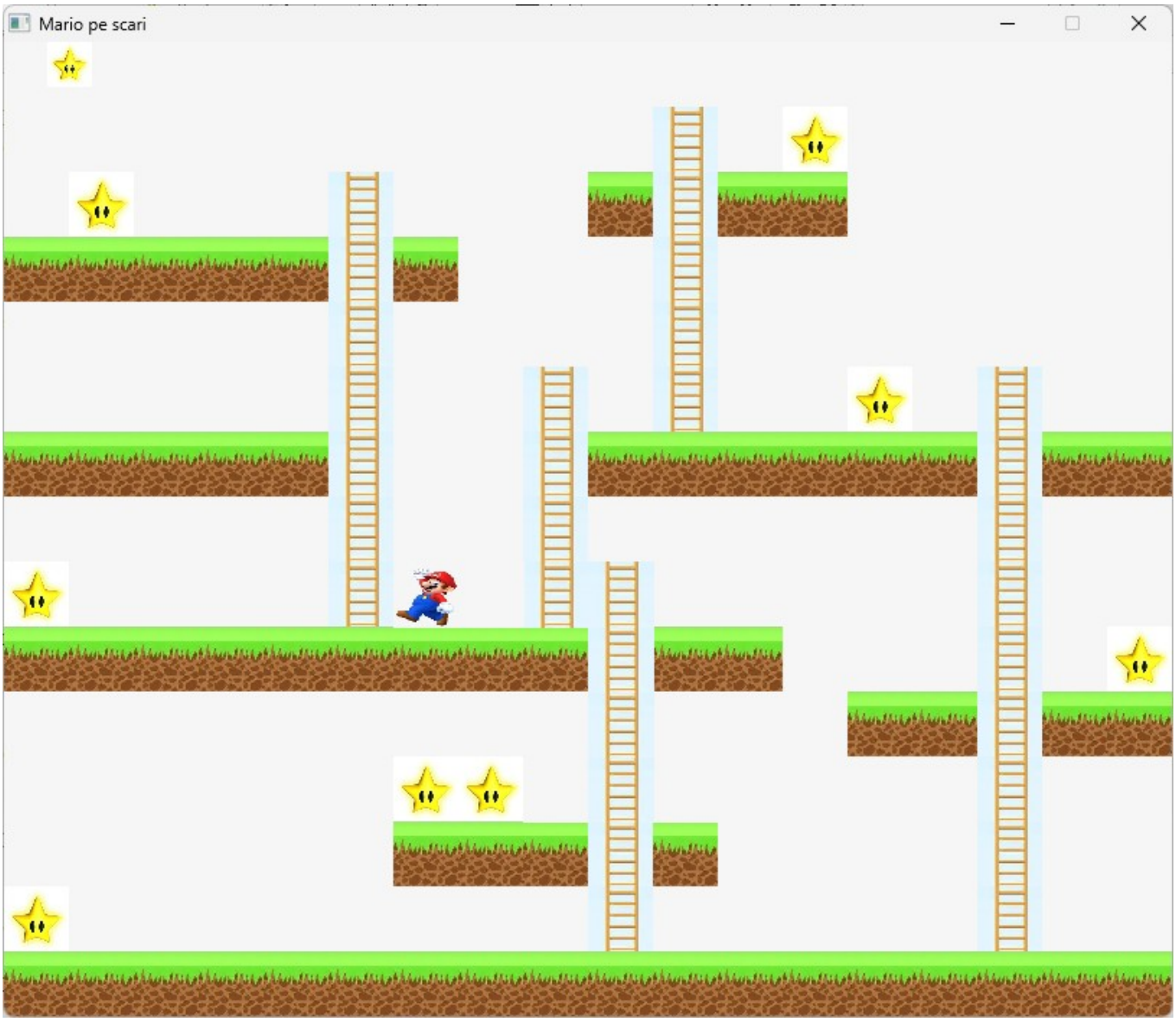
```
#include <winbgim.h>
#include <fstream>
#include <windows.h>
#include <time.h>

#define SUS 72
#define JOS 80
#define STG 75
#define DRP 77
#define ESC 27
#define SPC 32
#define TAB 9
#define BKS 8

#define DIR_STG 0
#define DIR_DRP 1
#define DIR_SUS 2
#define DIR_JOS 3

using namespace std;

int latime=50;

int n,ipisica,jpisica,isoarece,jsoarece;

int ipostaza_pisica, ipostaza_soarece;

int directie_soarece, directie_pisica;

char car, tasta;

char lab[20][20];

void deseneaza(char car, int i, int j)
{
    if (car=='1')
        readimagefile("brick.jpg",latime*j,latime*i,latime*(j+1),latime*(i+1));
    if (car=='0')
        readimagefile("blank.jpg",latime*j,latime*i,latime*(j+1),latime*(i+1));
}

void deseneazaPisica()
{
```

```
    to_string(ipostaza_pisica);
    string nume_poza="pisica"+to_string(ipostaza_pisica)+".jpg";
    readimagefile(nume_poza.c_str(),latime*jpisica,latime*ipisica,
                  latime*(jpisica+1),latime*(ipisica+1));
}

void deseneazaSoarece()
{
    to_string(ipostaza_soarece);
    string nume_poza="soarece"+to_string(ipostaza_soarece)+".jpg";

readimagefile(nume_poza.c_str(),latime*jsoarece,latime*isoarece,latime*(jsoarece+1),
latime*(isoarece+1));
}

void stergeSoarece()
{
    deseneaza(lab[isoarece][jsoarece],isoarece,jsoarece);
}

void stergePisica()
{
    deseneaza(lab[ipisica][jpisica],ipisica,jpisica);
}

void citesteLabirintul()
{
    ifstream fisier("labirint.txt");
    fisier>>n>>ipisica>>jpisica>>isoarece>>jsoarece;
    int i,j;
    for (i=0;i<n;i++)
        for (j=0;j<n;j++)
            {
                fisier>>car;
                lab[i][j]=car;
            }
    fisier.close();
}

void deseneazaLabirintul()
{
    int i,j;
    for (i=0;i<n;i++)
        for (j=0;j<n;j++)
            deseneaza(lab[i][j],i,j);
}

void stabilesteDirectiaSoarece()
{
    directie_soarece=rand()%4; // 0 stanga, 1 dreapta, 2 sus, 3 jos
    switch(directie_soarece) {
        case DIR_STG: if (jsoarece>0 && lab[isoarece][jsoarece-1]=='0') jsoarece--;
break;
        case DIR_DRP: if (jsoarece<n-1 && lab[isoarece][jsoarece+1]=='0') jsoarece+
+; break;
```

```
        case DIR_SUS: if (isoarece>0 && lab[isoarece-1][jsoarece]=='0') isoarece--;
break;
        case DIR_JOS: if (isoarece<n-1 && lab[isoarece+1][jsoarece]=='0') isoarece+
+; break;
    }
    //getch();
}

void stabilesteDirectiaPisica()
{
    if (kbhit())
    {
        tasta=getch();
        if (tasta==ESC) { closegraph(); exit(1); }
        if (tasta==0)
        {
            tasta=getch();
            switch(tasta) {
                case STG: directie_pisica=DIR_STG; break;
                case DRP: directie_pisica=DIR_DRP; break;
                case SUS: directie_pisica=DIR_SUS; break;
                case JOS: directie_pisica=DIR_JOS; break;
            }
            switch(directie_pisica) {
                case DIR_STG:
                  if (jpisica>0 && lab[ipisica][jpisica-1]=='0') jpisica--; break;
                case DIR_DRP:
                  if (jpisica<n-1 && lab[ipisica][jpisica+1]=='0') jpisica++; break;
                case DIR_SUS:
                  if (ipisica>0 && lab[ipisica-1][jpisica]=='0') ipisica--; break;
                case DIR_JOS:
                  if (ipisica<n-1 && lab[ipisica+1][jpisica]=='0') ipisica++; break;
            }
        }
    }
}

void schimbaIpostazaSoarece()
{
    switch(directie_soarece) {
        case DIR_SUS:
            if (ipostaza_soarece!=1) ipostaza_soarece=1; else ipostaza_soarece=2;
      break;
        case DIR_JOS:
            if (ipostaza_soarece!=3) ipostaza_soarece=3; else ipostaza_soarece=3;
      break;
        case DIR_STG:
            if (ipostaza_soarece!=7) ipostaza_soarece=7; else ipostaza_soarece=8;
      break;
        case DIR_DRP:
            if (ipostaza_soarece!=5) ipostaza_soarece=5; else ipostaza_soarece=6;
      break;
    }
}
```

```
void schimbaIpostazaPisica()
{
    switch(directie_pisica) {
        case DIR_SUS:
            if (ipostaza_pisica!=1) ipostaza_pisica=1; else ipostaza_pisica=2;
      break;
        case DIR_JOS:
            if (ipostaza_pisica!=3) ipostaza_pisica=3; else ipostaza_pisica=4;
      break;
        case DIR_STG:
            if (ipostaza_pisica!=7) ipostaza_pisica=7; else ipostaza_pisica=8;
      break;
        case DIR_DRP:
            if (ipostaza_pisica!=5) ipostaza_pisica=5; else ipostaza_pisica=6;
            break;
    }
}

int main()
{
    srand(time(NULL)); citesteLabirintul();
    initwindow(latime*n,latime*n,"Tom si Jerry in labirint",300,200);
    PlaySound("tomandjerry.wav",NULL,SND_ASYNC | SND_LOOP);
    readimagefile("tomjerry.jpg",0,0,latime*n,latime*n);
    getch(); deseneazaLabirintul();
    directie_pisica=DIR_STG; deseneazaPisica();
    ipostaza_soarece=1; deseneazaSoarece();
    do {
        stergeSoarece();
        //if (rand()%10<3)
        stabilesteDirectiaSoarece();
        for (int i=1;i<=2;i++)
            {
                schimbaIpostazaSoarece();
                deseneazaSoarece();
                delay(10);
            }
        //getch(); // delay(100);
        stergePisica(); stabilesteDirectiaPisica();
        schimbaIpostazaPisica(); deseneazaPisica();
        if (ipisica==isoarece && jpisica==jsoarece)
            {
                PlaySound("tomlaughing.wav",NULL,SND_SYNC);
                delay(5000);
                closegraph();
                exit(0);
            }
        delay(50);
    } while (tasta!=ESC);
    getch(); closegraph();
    return 0;
}
```

6.6. Mario pe scări



Fișierele necesare:

15 18 ....#.*..... .*...#...@#@@..... @@@@@#@...#..... .....#....#..... .....#..#.#..*.#.. @@@@@#..#@@@@@@#@@ .....#..#.....#.. *....#.*##.....#.. @@@@@@@@@#@@...#.* .....#...@@#@@ .....**.#M....#.. .....@@@#@....#.. *.....#.....#.. @@@@@@@@@@@@@@@@@@ **harta.txt**	 scara.jpg iarba.jpg fundal.jpg

mario1.jpg

mario2.jpg

mario3.jpg

mario4.jpg

mario5.jpg

mario6.jpg

mario_spate_1.jpg

mario_spate_2.jpg

stea.jpg

Codul complet al programului

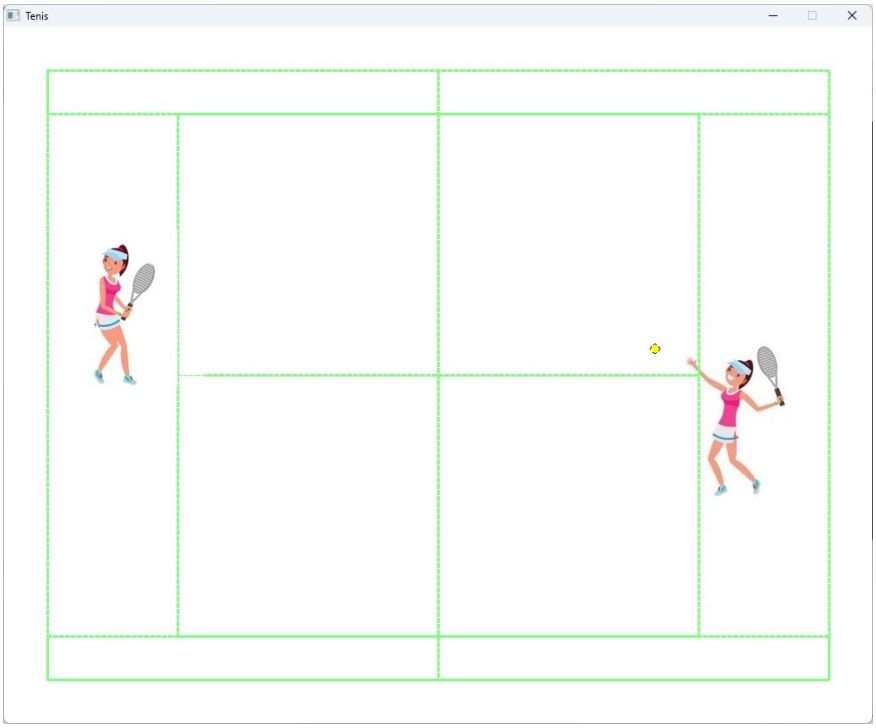
```
#include <fstream>
#include <winbgim.h>
#include <windows.h>
#include <string.h>

using namespace std;

#define MAX 20
#define latime 45

#define SUS 72
#define JOS 80
#define STG 75
#define DRP 77
#define ESC 27
#define SPC 32
#define TAB 9
#define BKS 8

int imario, jmario, ipostaza;
```

```
char tasta;
string directie;

int scor,nrstelute;

int nrLinii,nrColoane,i,j;
char harta[MAX][MAX];
char car;

ifstream fisier("harta.txt");

void afiseazaMario()
{
    int y=imario*latime, x=jmario*latime;

    switch(ipostaza) {
        case -10: readimagefile("mario_spate_1.jpg", x,y,x+latime,y+latime); break;
        case 10: readimagefile("mario_spate_2.jpg", x,y,x+latime,y+latime); break;
        case 1:
            readimagefile("mario1.jpg", x,y,x+latime,y+latime); break;
        case 2:
            readimagefile("mario2.jpg", x,y,x+latime,y+latime); break;
        case 3:
            readimagefile("mario3.jpg", x,y,x+latime,y+latime); break;
        case -1:
            readimagefile("mario4.jpg", x,y,x+latime,y+latime); break;
        case -2:
            readimagefile("mario5.jpg", x,y,x+latime,y+latime); break;
        case -3:
            readimagefile("mario6.jpg", x,y,x+latime,y+latime); break;
    }
}

void afiseazaPoza(char c, int i, int j)
{
    if (c=='@')
        readimagefile("iarba.jpg",latime*j,latime*i,latime*(j+1),latime*(i+1));
    if (c=='#')
        readimagefile("scara.jpg",latime*j,latime*i,latime*(j+1),latime*(i+1));
    if (c=='*')
        readimagefile("stea.jpg",latime*j,latime*i,latime*(j+1),latime*(i+1));
    if (c=='.')
        readimagefile("fundal.jpg",latime*j,latime*i,latime*(j+1),latime*(i+1));
}

void stergeMario()
{
    afiseazaPoza(harta[imario][jmario],imario,jmario);
}

void afiseazaScor()
{
    int i;
    for (i=1;i<=scor;i++)
        readimagefile("stea.jpg",30*i,0,30*i+30,30);
```

```
    afiseazaMario();
    if (scor<nrstelute)
    {
        for (i=800;i<=1200;i+=30) Beep(i,20);
    }
    else
    {
        PlaySound("7.wav",NULL,SND_SYNC);
        PlaySound("8.wav",NULL,SND_SYNC);
        closegraph();
        exit(0);
    }
}

void urmatoareaIpostaza()
{
    if (directie=="dreapta")
        {
            if (abs(ipostaza)==10) ipostaza=1;
            if (ipostaza<0) ipostaza=-ipostaza;
            ipostaza++; if (ipostaza==4) ipostaza=1;
            if (jmario<nrColoane-1 && harta[imario+1][jmario+1]!='.')
            {
                jmario++;
                if (harta[imario][jmario]=='*')
                {
                    scor++; harta[imario][jmario]='.'; afiseazaScor();
                }
            }
        }
    else
    if (directie=="stanga")
        {
            if (abs(ipostaza)==10) ipostaza=-1;
            if (ipostaza>0) ipostaza=-ipostaza;
            ipostaza--; if (ipostaza==-4) ipostaza=-1;
            if (jmario>0)
            {
                jmario--;
                if (harta[imario][jmario]=='*')
                {
                    scor++; harta[imario][jmario]='.'; afiseazaScor();
                }
            }
        }
    if (directie=="sus")
    {
        if (abs(ipostaza)==10) ipostaza=-ipostaza;
        else ipostaza=10;
        if (harta[imario-1][jmario]=='#') imario--;
    }
    if (directie=="jos")
    {
        if (abs(ipostaza)==10) ipostaza=-ipostaza;
        else ipostaza=-10;
```

```
        if (harta[imario+1][jmario]=='#') imario++;
    }
}

void incarcaHarta()
{
    fisier>>nrLinii>>nrColoane;
    initwindow(latime*nrColoane,latime*nrLinii,"Mario pe scari");
    setbkcolor(WHITE); cleardevice();
    nrstelute=0;
    for (i=0;i<nrLinii;i++)
    {
        for (j=0;j<nrColoane;j++)
            {
                fisier>>car;
                if (car=='*') nrstelute++;
                if (car=='M') {imario=i; jmario=j; car='.';}
                harta[i][j]=car;
                afiseazaPoza(harta[i][j],i,j);
            }
    }
    fisier.close();

    scor=0;
}

int main()
{
    incarcaHarta();
    ipostaza=1;
    afiseazaMario();
    //PlaySound("SuperMarioBros.wav", NULL, SND_ASYNC);
    // directie="dreapta";
    do
    {
        tasta=getch(); if (tasta==0) tasta=getch();
        if (tasta==STG && jmario>0 && harta[imario+1][jmario-1]!='.')
directie="stanga";
        if (tasta==DRP && jmario<nrColoane-1 && harta[imario+1][jmario+1]!='.')
            directie="dreapta";
        if (tasta==SUS && harta[imario-1][jmario]=='#') directie="sus";
        if (tasta==JOS && harta[imario+1][jmario]=='#') directie="jos";
        stergeMario();
        urmatoareaIpostaza();
        afiseazaMario();
        delay(200);
    } while (tasta!=ESC);  // tasta Esc (escape)

    closegraph();

    return 0;
}
```

6.7. Tenis




```
#include <time.h>
#include <iostream>
#include <winbgim.h>
#include <string>

#define STG 75
#define DRP 77
#define SUS 72
#define JOS 80
#define ESC 27

using namespace std;

string centrarea1;
string centrarea2;
int pozitia[3];

char tasta, tasta2;

int x[3]= {0,50,750};
int y[3]= {0,300,300};
int xbila, ybila;
int vxbila, vybila;

void afiseazaTeren()
{
    setcolor(LIGHTGREEN);
    setlinestyle(SOLID_FILL,0,3);
    rectangle(50,50,950,750);
    rectangle(50,50,950,100);
```

```
    rectangle(50,700,950,750);
    rectangle(200,100,800,700);
    line(500,100,500,700);
    line(200,400,800,400);
    setlinestyle(SOLID_FILL,0,1);
}

void stergeBila()
{
    setcolor(WHITE);
    setfillstyle(SOLID_FILL,WHITE);
    bar(xbila-7,ybila-7,xbila+7,ybila+7);
}

void afiseazaBila()
{
    setcolor(BLUE);
    setfillstyle(SOLID_FILL,YELLOW);
    fillellipse(xbila,ybila,6,6);
}

void miscaBila()
{
    delay(50);
    stergeBila();
    if (xbila<=x[1]+160 || xbila>=x[2]-10+30) { vxbila=-vxbila;
PlaySound("sunetbila.wav",NULL,SND_ASYNC);}
    if (vxbila==+1 && xbila>=500) { vybila=(ybila>y[2]?-1:1); }
    if (vxbila==-1 && xbila<500) vybila=(ybila>y[1]?-1:1);
    //if (ybila<=y[1] || ybila>=y[2]) { vybila=-vybila; }

    xbila=xbila+25*vxbila;
    ybila=ybila+10*vybila;

    afiseazaBila(); delay(30);
}

void stergeFata(int nrFata)
{
    readimagefile("spatiu.jpg",x[nrFata],y[nrFata],x[nrFata]+180,y[nrFata]+180);
}

void afiseazaFata(int nrFata)
{
    string s;
    if (nrFata==1)
        s="A"+centrarea1+to_string(pozitia[1])+".jpg";
    else
        s="B"+centrarea2+to_string(pozitia[2])+".jpg";
    //cout<<s<<endl;
    readimagefile(s.c_str(),x[nrFata],y[nrFata],x[nrFata]+180,y[nrFata]+180);
    /*
```

```
    if (s=="Adreapta1.jpg" || s=="Adreapta2.jpg" || s=="Adreapta3.jpg")
{ stergeBila(); delay(30); }
    if (s=="Bdreapta1.jpg" || s=="Bdreapta2.jpg" || s=="Bdreapta3.jpg")
{ stergeBila(); delay(30); }
    if (s=="Astanga2.jpg" || s=="Astanga3.jpg") { stergeBila(); delay(30); }
    if (s=="Bstanga2.jpg" || s=="bstanga3.jpg") { stergeBila(); delay(30); }
        */
    // if (nrFata==2) delay(200);
    afiseazaTeren();
    delay(80);
}

void sus(int nrFata, int distanta)
{
    if (y[nrFata]>distanta+50) y[nrFata]-=distanta;
}

void jos(int nrFata, int distanta)
{
    if (y[nrFata]<800-200-distanta) y[nrFata]+=distanta;
}

void schimbaCentrarea(int nrFata, int mod)
{
    int distanta=20+rand()%200;
    if (nrFata==1)
    {
        if (mod==SUS)
        {
            if (centrarea1=="centru")
            {
                stergeFata(nrFata);
                centrarea1="dreapta";
                sus(nrFata, distanta);
                afiseazaFata(nrFata);
            }
            else if (centrarea1=="stanga")
            {
                centrarea1="centru";
                afiseazaFata(nrFata);
            }
        }
        if (mod==JOS)
        {
            if (centrarea1=="centru")
            {
                stergeFata(nrFata);
                centrarea1="stanga";
                jos(nrFata, distanta);
                afiseazaFata(nrFata);
            }
            else if (centrarea1=="dreapta")
            {
```

```
                centrarea1="centru";
                afiseazaFata(nrFata);
            }
        }
    }
    else
    {
        if (mod==SUS)
        {
            if (centrarea2=="centru")
            {
                stergeFata(nrFata);
                centrarea2="dreapta";
                y[nrFata]-=20;
                afiseazaFata(nrFata);
            }
            else if (centrarea2=="stanga")
            {
                centrarea2="centru";
                afiseazaFata(nrFata);
            }
        }
        if (mod==JOS)
        {
            if (centrarea2=="centru")
            {
                stergeFata(nrFata);
                centrarea1="stanga";
                y[nrFata]+=20;
                afiseazaFata(nrFata);
            }
            else if (centrarea2=="dreapta")
            {
                centrarea2="centru";
                afiseazaFata(nrFata);
            }
        }
    }
}

void schimbaPozitia(int nrFata, int dir)
{
    if (dir==STG)
    {
        pozitia[nrFata]=pozitia[nrFata]-1;
        if (pozitia[nrFata]==0) pozitia[nrFata]=3;
        afiseazaFata(nrFata);
    }
    if (dir==DRP)
    {
        pozitia[nrFata]=pozitia[nrFata]+1;
        if (pozitia[nrFata]==4) pozitia[nrFata]=1;
        afiseazaFata(nrFata);
```

```
    }
}

void miscaFata1()
{
/*    if (kbhit())
    {
        tasta=getch();
        if (tasta==0)
        {
            tasta=getch();
        switch(tasta)
        {
        case SUS:
            schimbaCentrarea(1,SUS);
            break;
        case JOS:
            schimbaCentrarea(1,JOS);
            break;
        case STG:
            schimbaPozitia(1,STG);
            break;
        case DRP:
            schimbaPozitia(1,DRP);
            break;
        }
        }
        //cout<<centrarea<<" "<<pozitia<<endl;
    }
*/
    int distanta=20+rand()%200;
    if (rand()%100>30) return;
/*    if (xbila>500)
        {
            if (rand()%2==0) { stergeFata(1); sus(1,distanta); afiseazaFata(1);
return; }
        } */
    int t=rand()%4;
    switch(t) {
        case 0: tasta=STG; break;
        case 1: tasta=DRP; break;
        case 2: tasta=SUS; break;
        case 3: tasta=JOS; break;
    }
    switch(tasta) {
        case SUS:
            schimbaCentrarea(1,SUS);
            break;
        case JOS:
            schimbaCentrarea(1,JOS);
            break;
        case STG:
            schimbaPozitia(1,STG);
```

```
            break;
        case DRP:
            schimbaPozitia(1,DRP);
            break;
    }


}

void miscaFata2()
{
    int distanta=20+rand()%200;
    if (rand()%100>30) return;
/*    if (xbila<500)
        {
            if (rand()%2==0) { stergeFata(2); sus(2,distanta); afiseazaFata(2);
return; }
        } */
    int t=rand()%4;
    switch(t) {
        case 0: tasta2=STG; break;
        case 1: tasta2=DRP; break;
        case 2: tasta2=SUS; break;
        case 3: tasta2=JOS; break;
    }
    switch(tasta2) {
        case SUS:
            schimbaCentrarea(2,SUS);
            break;
        case JOS:
            schimbaCentrarea(2,JOS);
            break;
        case STG:
            schimbaPozitia(2,STG);
            break;
        case DRP:
            schimbaPozitia(2,DRP);
            break;
    }
}

int main()
{
    srand(time(NULL));
    initwindow(1000,800,"Tenis");
    setbkcolor(WHITE);
    cleardevice();
    centrarea1="centru";
    pozitia[1]=1;
    centrarea2="centru";
    pozitia[2]=3;
    afiseazaFata(1);
    afiseazaFata(2);
```

```
    vxbila=1; vybila=1;
    xbila=x[1]+200; ybila=y[1]+50;
    afiseazaBila();

    do
    {
        miscaFata1();
        miscaFata2();
        miscaBila();
    }
    while (tasta!=ESC);
    return 0;
}
```

Capitolul 7. Aplicații pentru tablouri bi- și tridimensionale (II)

7.1. Pirates Hide and Seek

https://youtu.be/jnwyCrX586A

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

7.2. Gobblet Gobblers

https://www.youtube.com/watch?v=CwG3LMaMOZA

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

Demo cu doua piese:

```
/*
initwindow(latime,inaltime,titlu fereastra)
closegraph()
line(x1,y1,x2,y2)
rectangle(x1,y1,x2,y2)
readimagefile("fisier.jpg",x1,y1,x2,y2)
kbhit() = true daca chiar atunci a fost apasata o tasta, false in caz contrar
ismouseclick(WM_LBUTTONDOWN) = true daca chiar atunci a fost apasat butonul Left
(stanga)
if(ismouseclick(WM_LBUTTONDOWN))
      {
      clearmouseclick(WM_LBUTTONDOWN);
      x=mousex();
      y=mousey();
      ....
      }
*/

#include <iostream>
#include <winbgim.h>

using namespace std;

int width,height,latura,numar;
int jucator;

int TablaDeJoc[20][20];

void punerePiesa()
{
      int linia,coloana,x,y;
      int x1, y1, x2, y2;
      int xmijloc, ymijloc;
      if(ismouseclick(WM_LBUTTONDOWN))
      {
      clearmouseclick(WM_LBUTTONDOWN);
      x=mousex();
      y=mousey();
      linia=y/latura+1;
```

```
      coloana=x/latura+1;
      if (TablaDeJoc[linia][coloana]==0)
      {
            jucator=3-jucator;
            TablaDeJoc[linia][coloana]=jucator;
            x1=latura*(coloana-1);
            y1=latura*(linia-1);
            x2=x1+latura;
            y2=y1+latura;
            cout<<"Pun piesa jucatorului "<<jucator<<" pe linia "<<linia<<" si
coloana "<<coloana<<endl;
            if (jucator==1)
                  readimagefile("1.jpg",x1,y1,x2,y2);
            else
                  readimagefile("2.jpg",x1,y1,x2,y2);
      }
      }
}

void desen()
{
      int i,j;
      rectangle(0,0,width-1,height-1);
      for (i=1; i<=numar; i++)
      for (j=1; j<=numar; j++)
            rectangle(latura*(i-1),latura*(j-1),latura*i,latura*j);
}

int main()
{
      numar=10;
      width=700;
      height=width;
      latura=width/numar;
      initwindow(width,height,"Demo");
      desen();
      jucator=1;
      do
      {
      punerePiesa();
      }
      while (!kbhit());
      closegraph();
      return 0;
}
```

Amplasăm piese:

```
#include <iostream>
#include <winbgim.h>

#define lat 100
#define inalt 200
```

```
using namespace std;


struct stiva
{
	int p[4];
	int nr;
};

stiva T[4][4];

void deseneazaCelula(int linia, int coloana)
{
	// trebuie sa desenez continutul celulei (linia,coloana) din T
	int varf=T[linia][coloana].nr;
	switch (T[linia][coloana].p[varf])
	{
	case -1:
	readimagefile("1p.jpg", lat*(coloana-1),inalt*(linia-
1),lat*coloana,inalt*linia);
	break;
	case -2:
	readimagefile("2p.jpg", lat*(coloana-1),inalt*(linia-
1),lat*coloana,inalt*linia);
	break;
	case -3:
	readimagefile("3p.jpg", lat*(coloana-1),inalt*(linia-
1),lat*coloana,inalt*linia);
	break;
	case 1:
	readimagefile("1a.jpg", lat*(coloana-1),inalt*(linia-
1),lat*coloana,inalt*linia);
	break;
	case 2:
	readimagefile("2a.jpg", lat*(coloana-1),inalt*(linia-
1),lat*coloana,inalt*linia);
	break;
	case 3:
	readimagefile("3a.jpg", lat*(coloana-1),inalt*(linia-
1),lat*coloana,inalt*linia);
	break;
	}
	setlinestyle(0,0,3);
	setcolor(LIGHTGREEN);
	line(lat,0,lat,3*inalt);
	line(2*lat,0,2*lat,3*inalt);
	setcolor(LIGHTRED);
	line(0,inalt,3*lat,inalt);
	line(0,2*inalt,3*lat,2*inalt);
}

void punePiesa(int linia, int coloana, int x)
{
	if (T[linia][coloana].nr==3)
```

```
	cout<<"Nu se poate amplasa, deoarece sunt deja 3 piese.";
	else
	{
	int varf=T[linia][coloana].nr;
	int y=T[linia][coloana].p[varf];
	if (abs(x)<=abs(y))
		cout<<"Nu se poate amplasa decat o piesa mai mare peste o alta piesa.";
	else
	{
		T[linia][coloana].nr++;
		T[linia][coloana].p[T[linia][coloana].nr]=x;
		deseneazaCelula(linia,coloana);
	}
	}
}



void deseneazaTabla()
{
	int linia,coloana;
	for (linia=1; linia<=3; linia++)
	for (coloana=1; coloana<=3; coloana++)
		deseneazaCelula(linia,coloana);
}


int main()
{
	initwindow(lat*3,inalt*3,"Gobblet Gobblers",300,200);
	setbkcolor(WHITE); cleardevice();
	int sir[20]={0,-1,3,-2,3,-3,2};
	int lin[20]={0,1,1,1, 2, 3,1};
	int col[20]={0,1,1,2, 2, 1,2};

	deseneazaTabla();
	for (int i=1;i<=6;i++)
	{
	cout<<"Amplasam piesa "<<sir[i]<<" pe linia "<<lin[i]<<" si coloana
"<<col[i]<<endl;
	punePiesa(lin[i],col[i],sir[i]);
	getch();
	}
	getch();
	closegraph();
	return 0;
}
```

7.3. Smart Farmer (varianta simplă / cu FILL)

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

Capitolul 8. Aplicații liste, stive și cozi (II)

8.1. Calculator în limba română

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

8.2. Graficul unei funcții

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

Capitolul 9. Paradigme de elaborare a algoritmilor

9.1. Greedy

9.2. Divide et impera

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

9.3. Programare dinamică

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

```
#include <iostream>

using namespace std;

int main()
{
  int n,nr[100],minimul,i;
  cout << "Dati numarul n" << endl;
  cin>>n;
  nr[1]=0;
  nr[2]=1;
  nr[3]=1;
  for(i=4; i<=n; i++)
  {
    if(i%3==0 && i%2==0 )
    {
      minimul=nr[i-1];
      if(nr[i/3]<minimul)minimul=nr[i/3];
      if(nr[i/2]<minimul)minimul=nr[i/2];
      nr[i]=minimul+1;
    }
    else if(i%3==0 && i%2!=0 )
    {
      minimul=nr[i-1];
      if(nr[i/3]<minimul)minimul=nr[i/3];
      nr[i]=minimul+1;

    }
    else if(i%3!=0 && i%2==0 )
    {
      minimul=nr[i-1];
      if(nr[i/2]<minimul)minimul=nr[i/2];
      nr[i]=minimul+1;
    }
    else
    {
      minimul=nr[i-1];
      nr[i]=minimul+1;
    }
```

```
  }
  cout<<nr[n];
  return 0;
}
```

9.4. Backtracking

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

```
#include <iostream>

using namespace std;

int n;

int nr_solutie=0;

int x[10];

void afisez_solutia()
{
  nr_solutie++;
  cout<<"\nSolutia numarul "<<nr_solutie<<endl;
  for (int i=1;i<=n;i++)
    cout<<"Afiseaza regina nr. "<<i<<" pe coloana "<<i<<" si linia
"<<x[i]<<endl;
}

bool pot_continua(int k)
{
  for (int i=1;i<k;i++) // verific daca regina k ataca vreo regina anterioara
    {
      if (x[i]==x[k]) return false; // aceeasi linie
      if (abs(x[i]-x[k])==k-i) return false; // atac pe diagonala
    }
  return true;
}

int main()
{
  cout<<"n="; cin>>n;
  bool buna;
  int k; // regina curenta, coloana curenta
  k=1; x[1]=0; // pun regina nr. 1 sub tabla (linia 0)
  while (k>0) // cat timp mai sunt regine pe tabla
  {
    buna=false;
    while (x[k]<n && !buna) // cat timp poti muta regina mai sus si pozitia nu e
buna
      {
        x[k]++; // mut regina mai sus
        if (pot_continua(k)) buna=true;
      }
    if (buna) // daca am gasit o linie buna, pt. a aseza regina
```

```
    {
      if (k<n)
      {
        //trec la regina urmatoare
        k++;
        // asez regina urmatoare sub tabla
        x[k]=0;
      }
      else { // am ajuns la ultima regina
        afisez_solutia();
      }
    }
    else // am epuizat toate liniile si nu am (mai) gasit o linie buna
    { // ma intorc la regina anterioara
      k--;
    }
  }
  cout<<"\nTotal "<<nr_solutie<<" solutii.\n";
  return 0;
}
```

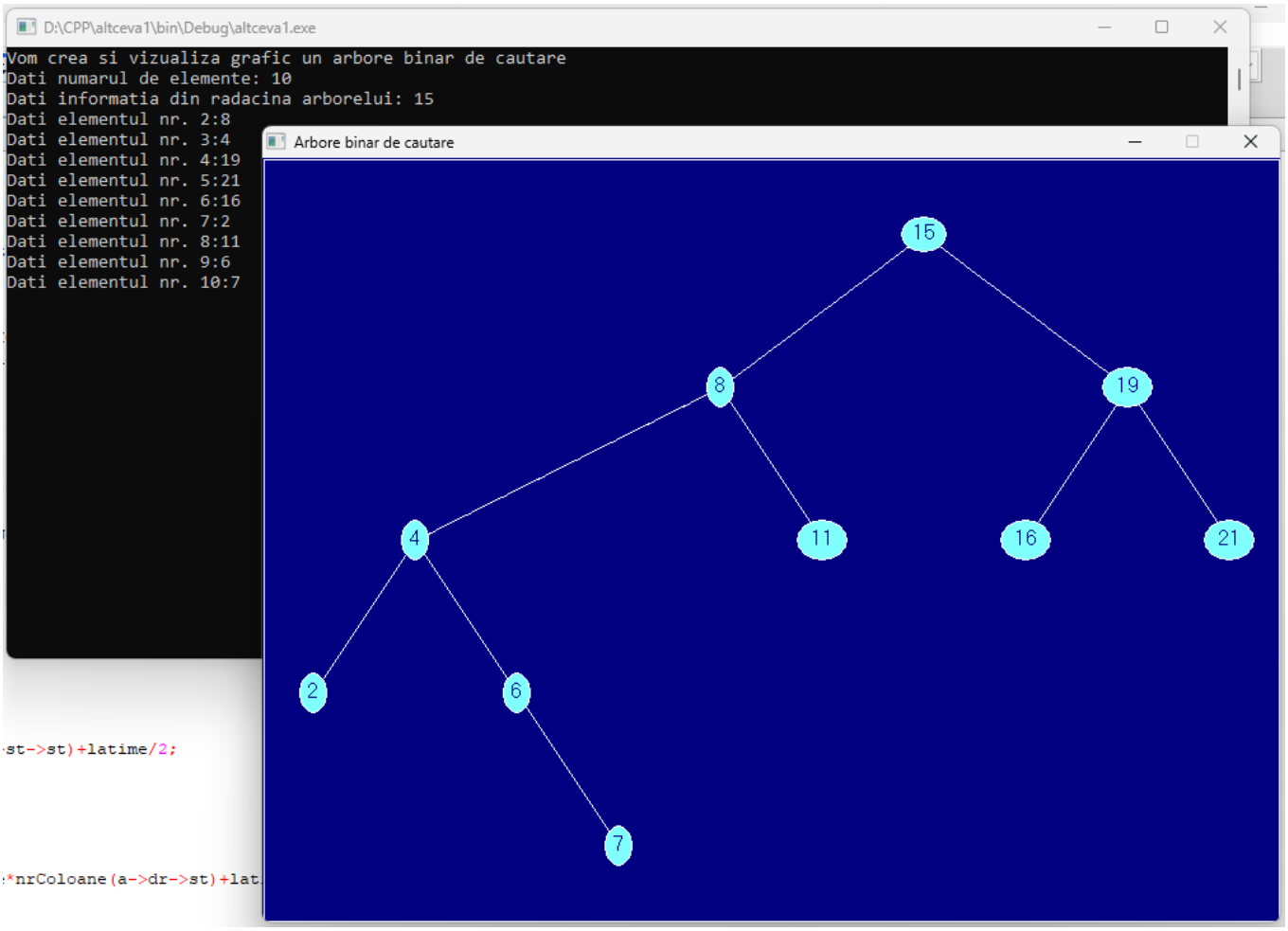
9.5. Branch and bound

ACEASTA SECȚIUNE ESTE ÎN LUCRU.

Clasa a XI-a

Capitolul 10. Aplicații structuri arborescente

10.1. Reprezentarea grafică a unui arbore binar de căutare



Următorul program va desena un arbore binar de căutare, în care întâi se citește informația din nodul rădăcină, apoi celelalte numere.

Programul nostru are următoarele funcții:

- Funcția "nrNiveluri" determină numărul de niveluri ale arborelui, adică lungimea celei mai lungi căi de la rădăcină la frunze. Funcția primește ca parametru un pointer la nodul rădăcină a arborelui și returnează numărul de niveluri.
- Funcția "nrColoane" determină numărul de coloane necesare pentru a desena arborele, adică numărul total de noduri din arbore. Funcția primește ca parametru un pointer la nodul rădăcină a arborelui și returnează numărul de coloane.
- Funcția "deseneazaNod" primește informația unui nod și coordonatele centrului nodului în fereastra de afișare, precum și dimensiunile nodului și

desenează un cerc cu centrul în coordonatele respective, umplut cu o culoare și afișează informația nodului în interiorul cercului.

- Funcția "deseneazaArbore" primește un pointer la nodul rădăcină a arborelui, nivelul curent al nodului în arbore, coordonata stângă a arborelui, lățimea și înălțimea unui nod și desenează arborele. Funcția folosește coordonatele pentru a plasa nodul curent în fereastra de afișare și apoi desenează liniile care conectează nodul curent cu părintele său și cu copiii săi. Funcția apoi desenează arborele recursiv pentru copiii nodului curent.
- Funcția "adaugaLaArboreElement" primește un pointer la nodul rădăcină a arborelui și un element de adăugat și îl adaugă în arbore. Funcția folosește regulile unui arbore binar de căutare pentru a determina locul corect de inserție a elementului în arbore.

Funcția "main" este funcția principală a programului. Aceasta întâi primește numărul de elemente pe care trebuie să le adauge în arbore și informatia elementului radacinii. Apoi apelează funcția "adaugaLaArboreElement" pentru a adăuga fiecare element în arbore. Funcția "main" desenează apoi arborele folosind funcția "deseneazaArbore". Dimensiunile nodurilor și distanța dintre ele sunt calculate din numărul de coloane și niveluri ale arborelui. La sfârșit, utilizatorul trebuie să apese o tastă pentru a închide fereastra de afișare și a termina programul.

Codul complet al programului

```
#include <iostream>
#include <winbgim.h>
#include <cstring>
#include <stdio.h>

using namespace std;
struct nod
{
    int info;
    nod *st, *dr;
};

int nrNiveluri(nod *a)
{
    if(a==NULL)return 0;
    else
    {
        int n1=nrNiveluri(a->st);
        int n2=nrNiveluri(a->dr);
        return 1+max(n1,n2);
    }
}
int nrColoane(nod *a)
{
    if(a==NULL)return 0;
    else
    {
```

```
         int n1=nrColoane(a->st);
        int n2=nrColoane(a->dr);
        return 1+n1+n2;
    }
}

void deseneazaNod(int elem, int xc, int yc, int latime, int inaltime)
{
    char arr[50];
    sprintf(arr,"%d",elem);
    setcolor(WHITE); setfillstyle(SOLID_FILL,LIGHTCYAN);
    fillellipse(xc,yc,textwidth(arr)+2, textheight("M")/2+6);
    settextstyle(SANS_SERIF_FONT,HORIZ_DIR,1);
    setbkcolor(LIGHTCYAN);
    setcolor(BLUE);
    settextjustify(1,1);
    outtextxy(xc,yc+4,arr);
}

void deseneazaArbore(nod *a, int niv, int stanga, int latime, int inaltime)
{
    if(a!=NULL)
    {
        int n1=nrColoane(a->st);
        int xc=stanga+latime*n1+latime/2;
        int yc=niv*inaltime-inaltime/2;

        if (a->st!=NULL)
        {
            int xcs=stanga+latime*nrColoane(a->st->st)+latime/2;
            setcolor(WHITE);
            line(xc,yc,xcs,yc+inaltime);
        }
        if (a->dr!=NULL)
        {
            int xcd=stanga+latime*(n1+1)+latime*nrColoane(a->dr->st)+latime/2;
            setcolor(WHITE);
            line(xc,yc,xcd,yc+inaltime);
        }
        deseneazaNod(a->info,xc,yc,latime, inaltime);
        deseneazaArbore(a->st,niv+1,stanga, latime, inaltime);
        deseneazaArbore(a->dr,niv+1,stanga+latime*(n1+1), latime, inaltime);
    }
}
void adaugaLaArboreElement(nod* &a, int el)
{
    if (a==NULL)
    {
        a=new nod;
        a->info = el;
        a->st = NULL;
        a->dr = NULL;
    }
    else if (el < a->info)
        adaugaLaArboreElement(a->st, el);
```

```
    else
        adaugaLaArboreElement(a->dr, el);
}


int main()
{
    nod *radacina;
    radacina=NULL;
    int nrelem,elem;
    cout<<"Vom crea si vizualiza grafic un arbore binar de cautare\n";
    cout<<"Dati numarul de elemente: ";
    cin>>nrelem;
    cout<<"Dati informatia din radacina arborelui: ";
    cin>>elem;
    adaugaLaArboreElement(radacina,elem);
    for(int i=2; i<=nrelem; i++)
    {
        cout<<"Dati elementul nr. "<<i<<":";
        cin>>elem;
        adaugaLaArboreElement(radacina,elem);
    }
    int height=600, width=800;
    initwindow(width,height,"Arbore binar de cautare");
    setcolor(WHITE); setbkcolor(BLUE); cleardevice();
    rectangle(1,1,width-1,height-1);
    int latime, inaltime;
    latime=width/nrColoane(radacina);
    inaltime=height/nrNiveluri(radacina);
    deseneazaArbore(radacina,1,0,latime,inaltime);
    getch(); closegraph();
    return 0;
}
```

10.2. Evaluator matematic și reprezentarea grafică a arborelui asociat expresiei

Acest cod C++ este un program care evaluează o expresie matematică, stocată element cu element într-un vector, într-un punct real x. Programul se bazează pe o structură de date de stivă, care este utilizată pentru a păstra operatorii și operanzii în timpul procesării expresiei. Sunt incluse, de asemenea, mai multe funcții matematice pentru a efectua operațiile corespunzătoare.

Structura de date "functie" este definită pentru a păstra informații despre expresie, inclusiv intervalul de evaluare și numărul de puncte de evaluare.

În funcția "ValoareFunctie", expresia este evaluată într-un mod similar cu evaluarea expresiilor matematice în mod tradițional, utilizând o stivă pentru a ține evidența operatorilor și operanzilor. În timpul procesării expresiei, funcțiile matematice sunt apelate pentru a efectua operațiile corespunzătoare.

De exemplu, funcția "Prioritate" este utilizată pentru a determina prioritatea operatorilor în funcție de simbolul operatorului. Funcția "EsteNumar" este utilizată pentru a determina dacă un element din vectorul expresiei este un număr.

Pentru a efectua evaluarea, expresia este parcursă element cu element și se fac mai multe verificări pentru a determina tipul elementului (operator sau operand) și pentru a efectua operațiile corespunzătoare. În timpul procesării expresiei, stiva operatorilor și stiva operanzilor sunt actualizate în consecință.

La sfârșit, valoarea finală a expresiei este stocată în variabila "valo" și este returnată de funcția "ValoareFunctie".

```
#include <math.h>
#include <string.h>
#include <limits.h>
#include <stdlib.h>
#include <ctype.h>
#define infinit INT_MAX
#define epsi 0.0001
#define MAX 100
#define MAX1 20
#include <stdio.h>

int top1,top2; // varfurile celor doua stive
const int max_stiva=100;

double Opd[max_stiva]; // stiva operanzilor
char Op[max_stiva]; // stiva operatorilor

char OperatiiBinare[200]="+-*/^<>=#";
char OperatiiUnare[200]="scarel";
char Operatii[200]="+-*/^<>=#scarel";

int Prioritate(char c)  // prioritate operatorilor
{
    if(c=='(' || c==')')
        return 0;
    if(c=='+' || c=='-')
        return 1;
    if(c=='*' || c=='/')
        return 2;
    if(c=='^')
        return 3;
    if(c=='=' || c=='#' || c=='<' || c=='>')
        return 4;
    if(c=='c' || c=='s' || c=='l' || c=='e' || c=='t' || c=='a' || c=='r')
        return 5;
}

const float pi=3.1415926; // se da sub forma literei q

bool DifInf(float x)
```

```
{
    return fabs(infinit-fabs(x)) > infinit / 2.0;
}

float Logaritm(float x)
{
    if (x>epsi && DifInf(x))
        return log(x);
    else
        return infinit;
}

float Exponential(float x)
{
    if (DifInf(x)) return exp(x);
    else return infinit;
}

float Inmultit(float x, float y)
{
    if (fabs(x < epsi) || fabs(y) < epsi) return 0;
        else if (DifInf(x) && DifInf(y)) return x*y;
            else return infinit;
}

float Putere(float x, float y)
{
    // float p;
    if (x==0) return 0;
    else if (y==0) return 1;
    else if (x==infinit || y==infinit) return infinit;
    else
        return pow(x,y);
}

float Egal(float x, float y)
{
    return x==y;
}

float Diferit(float x, float y)
{
    return x!=y;
}

float MaiMic(float x, float y)
{
    return x < y;
}

bool MaiMare(float x, float y)
{
    return x > y;
}
```

```
float Plus(float x, float y)
{
    if (DifInf(x) && DifInf(y))  return x+y;
    else return infinit;
}

float Minus(float x, float y)
{
    if (DifInf(x) && DifInf(y))  return x-y;
    else return infinit;
}

float Impartit(float x, float y)
{
    if (fabs(y)>epsi) return x/y;
    else return infinit;
}

float Sinus(float x)
{
    if (DifInf(x))  return sin(x);
    else return infinit;
}

float Cosinus(float x)
{
    if (DifInf(x))  return cos(x);
    else return infinit;
}

float Modul(float x)
{
    if (DifInf(x)) return fabs(x);
    else return infinit;
}

float Radical(float x)
{
    if (DifInf(x) && (x>epsi)) return sqrt(x);
    else return infinit;
}

bool EsteNumar(char sir[MAX1])
{
    return (atof(sir)!=0.0 &&
            strchr("0123456789",sir[0]));
}

struct functie
{
    char expresie[300];
    char vect[MAX+1][MAX1];
    // vectorul cuprinzand “cuvintele”ce formeaza expresia}
    int lung; // lungimea efectiva a vectorului
    float a,b; // intervalul de evaluare a functiei
```

```
    int n; //numarul de puncte de evaluare
};



void depanare()
{
    printf("Stiva operanzilor: ");
    for (int i=1; i<=top1; i++)
        printf("%f,",Opd[i]);
    printf("\nStiva operatorilor: ");
    for (int i=1; i<=top2; i++)
        printf("%c,",Op[i]);
    printf("\n");
}

double ValoareFunctie(functie E, double x)
{
    int i;
    // returneaza valoarea functiei E in punctul x
    double valo,x_1,x_2;
    //calculul expresiei
    for (i=1; i<=max_stiva; i++)
    {
        Opd[i]=0;
        Op[i]='@';
    }
    top1=0;
    top2=1;
    Op[top2]='(';
    i=0;
    while (i<=E.lung && top2>0)
    {
        i++;
        if (EsteNumar(E.vect[i]))
        {
            // printf("\nE.vect[%d]=%s\n",i,E.vect[i]);
            top1++;
            valo=atof(E.vect[i]);
            Opd[top1]=valo;
            // depanare();
        }
        else
            switch (E.vect[i][0]) {
            /* constanta pi=3.1415926 se da sub forma literei q */
            case 'q': top1++; Opd[top1]=pi; break;
            case 'X': /* variabila x */ top1++; Opd[top1]=x;
                /* printf("\n-->x=%f\n",x); */ break;
            case '(': /* inceput de bloc */ top2++; Op[top2]='('; break;
            default:
                /* operatii binare si unare */
               while ((top2>0) && !(strchr("()",Op[top2])) &&
                   Prioritate(Op[top2])>=Prioritate(E.vect[i][0]))
                {
                    if (top1>1) x_1=Opd[top1-1];
```

```
                    x_2=Opd[top1];
                    // depanare();
                    /* functii scrise in OPERATII */
                    switch (Op[top2]) {
                        case '=': valo=Egal(x_1,x_2); break;
                        case '#': valo=Diferit(x_1,x_2); break;
                        case '<': valo=MaiMic(x_1,x_2); break;
                        case '>': valo=MaiMare(x_1,x_2); break;
                        case '+': valo=Plus(x_1,x_2); break;
                        case '-': valo=Minus(x_1,x_2); break;
                        case '*': valo=Inmultit(x_1,x_2); break;
                        case '/': valo=Impartit(x_1,x_2); break;
                        case '^': valo=Putere(x_1,x_2); break;
                        case 's': valo=Sinus(x_2); break;
                        case 'c': valo=Cosinus(x_2); break;
                        case 'l': valo=Logaritm(x_2); break;
                        case 'e': valo=Exponential(x_2); break;
                        case 'a': valo=Modul(x_2); break;
                        case 'r': valo=Radical(x_2); break;
                    } //operator[top2]
                    if (strchr(OperatiiBinare,Op[top2])) top1--;
                    if (strchr(Operatii,Op[top2])) Opd[top1]=valo;
                    top2--;
                }
            // depanare();
            if (top2>0)
                if ((Op[top2]!='(') || (strcmp(E.vect[i],")")))
                    {
                        top2++; Op[top2] = E.vect[i][0];
                    }
                else top2--;
            }
    }
    if (top2==0) return Opd[1];
    else return infinit;
}

int main()
{
    functie F;
    //strcpy(F.expresie,"(X+2*X)/4");
    //strcat(F.expresie,")");
    //F.lung=strlen(F.expresie);

    strcpy(F.vect[0],"("); //!!!
    strcpy(F.vect[1],"(");
    strcpy(F.vect[2],"X");
    strcpy(F.vect[3],"+");
    strcpy(F.vect[4],"2");
    strcpy(F.vect[5],"*");
    strcpy(F.vect[6],"X");
    strcpy(F.vect[7],")");
    strcpy(F.vect[8],"/");
    strcpy(F.vect[9],"4");
    strcpy(F.vect[10],")");
```

```
    F.lung=11;
    float a=ValoareFunctie(F,3);
    printf("%f",a);
}
```

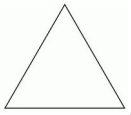
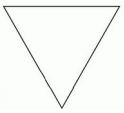
10.3. Derivare formală

Vezi videoclip aici https://youtu.be/UBaWrc-tChQ

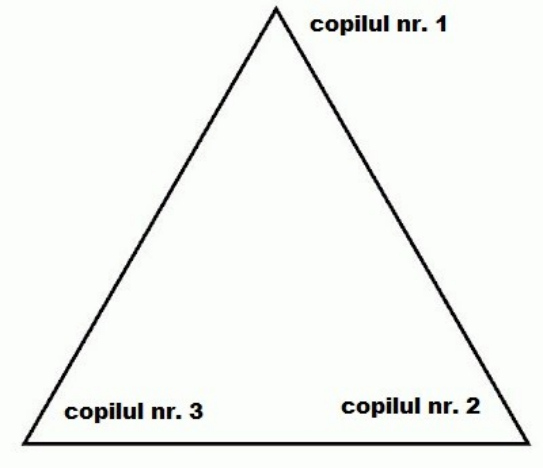
10.4. O problemă cu triunghiuri

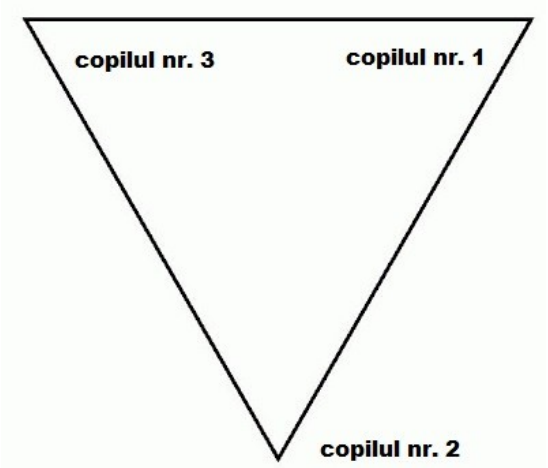
Problema următoare a fost dată la concursul de informatică pentru fete „The Beauty of Code - The Code of Beauty”, ed. a II-a, organizat Asociația EduSoft, în cadrul cercului de informatică „Programare cu răbdare”, în 2019

Se dă de la tastatură un șir de caractere ce conține doar caracterele: A, V, , , #, (și). A reprezintă un triunghi echilateral cu vârful sus și baza jos, iar V reprezintă un triunghi echilateral cu vârful jos și baza sus.

A reprezintă  , iar **V** reprezintă  .

Fiecare asemenea triunghi poate avea în colțurile sale alte triunghiuri „copii”, de tip **A** sau **V**, cu laturile jumătate din laturile triunghiului dat. Acest lucru este reprezentat, trecând între paranteze copiii, separați de virgulă. Copiii sunt trecuți în ordinea acelor de ceasornic, de la copilul 1 la copilul 3, conform desenelor și regulilor de mai jos:



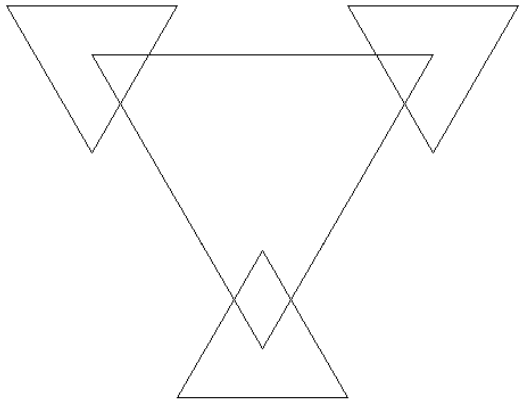
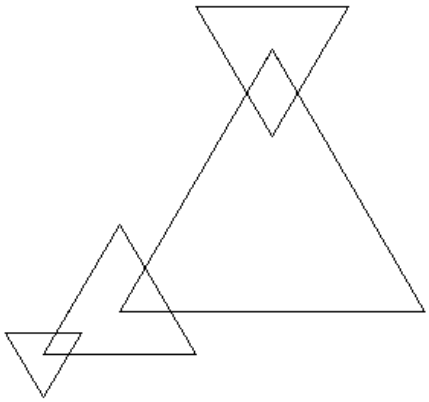
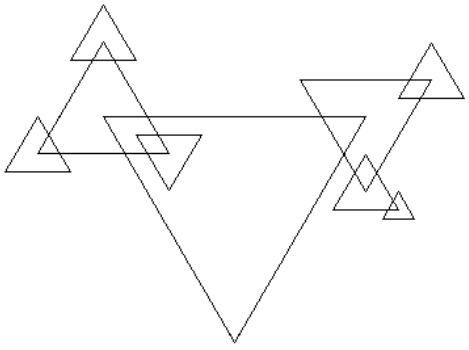


1. Când toți copiii lipsesc, nu se mai trec nici măcar parantezele.
2. Când toți copiii sunt prezenți, aceștia se trec între paranteze, separați de virgule.
3. Când copilul 1 este prezent, dar copiii 2 și 3 lipsesc, se trece între paranteze doar copilul 1.
4. Când copilul 2 este prezent, dar copiii 1 și 3 lipsesc, se trece # în locul primului copil, iar copilul 3 nu se mai trece.
5. Când copilul 3 este prezent, dar copiii 1 și 2 lipsesc, se trece # în locul primilor doi copil, iar copilul 3 se trece la final.
6. Când copii 1 și 2 sunt prezenți, dar copilul 3 lipsește, se trec între paranteze doar copiii 1 și 2.

7. Când copii 2 și 3 sunt prezenți, dar copilul 1 lipsește, se trece simbolul # în locul primului copil, apoi se trec ceilalți doi copii.
8. Când copiii 1 și 3 sunt prezenți, dar copilul 2 este absent, se trece simbolul # în locul celui de-al doilea copil.

Fiecare copil poate avea la rândul său copii și așa mai departe, pentru care se vor aplica regulile de mai sus, mai departe.

Iată câteva șiruri și reprezentările lor prin triunghiuri:

Șir de intrare	**Reprezentare grafică**
V(V,A,V) **Explicație:** triunghiul mare (de tip V) are trei copii, primul este de tip V, al doilea de tip A, iar al treilea de tip V.	
A(V,#,A(#,#,V)) **Explicație:** triunghiul mare (de tip A) are copilul 1 de tip V, copilul 2 lipsește (s-a trecut primul simbol #), apoi exista copilul 3 de tip A. Acesta, la rândul său, are doar copilul al treilea (V), restul copiilor fiind absenți (s-au trecut cele două simboluri # înaintea lui V).	
V(V(A,A(#,A)),#,A(A,V,A)) **Explicație:** triunghiul mare (de tip V) are copilul 1 (un V cu copiii A, A (cu copilul al doilea A, primul lipsind)), copilul 2 lipsește, iar copilul 3 este un A (cu 3 copii, de tip A, V și A)	

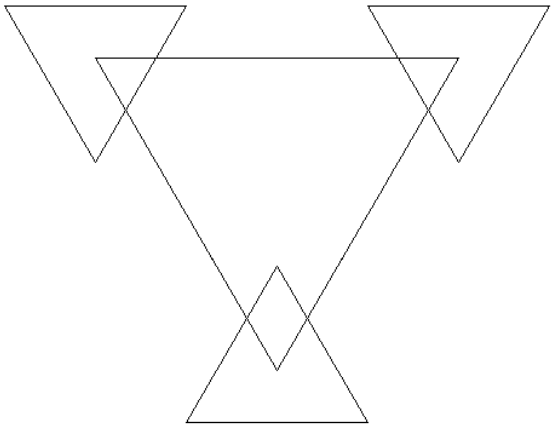
Problema cere ca plecând de la un asemenea șir de intrare să se obțină:

A) **pe ecran desenul corespunzător șirului**
B) **în fișierul TRIUNGHI.TXT coordonatele segmentelor de dreaptă care compun desenul.**

Precizări:

1) Șirul de intrare este corect definit și nu conține mai mult de 30 de caractere (A,V,virgula, paranteze, #). În loc de șir de caractere se poate folosi un vector cu numere (în care 1 specifică A, 2 reprezintă V, 3 reprezintă #, 4 reprezintă virgula, 5 reprezintă paranteza deschisă, iar 6 paranteza închisă).
2) Structura de date pe care o folosiți rămâne la latitudinea voastră.
3) Dacă algoritmul vostru nu rezolvă problema pe orice caz, dar rezolvă pe anumite cazuri, primiți punctaj parțial.
4) Fiecare triunghi poate avea copii, nepoți, strănepoți etc. până la cel mult nivelul 5 de urmași.
5) Pentru a realiza desenul, se va folosi *doar* biblioteca grafică WinBGIm, ce poate fi descărcată și instalată în CodeBlocks de la adresa: https://programarecurabdare.ro/grafica-in-codeblocks/
6) Desenarea se va face într-o fereastră grafică cu 800 de pixeli lățime și 600 pixeli înălțime
7) Primul triunghi va fi plasat în mijlocul ferestrei grafice (punctul de coordonate 400, 300). Latura primului triunghi va fi de 300 de pixeli.
8) Coordonatele segmentelor de dreaptă se trec în ordinea: abscisa primului punct, ordonata primului punct, abscisa celui de-al doilea punct, ordonata celui de-al doilea punct.
9) Fiecare segment va fi salvat în fișier pe o linie separată, sub forma a 4 numere naturale, separate de spațiu, reprezentând coordonatele menționate la punctul anterior.
10) Nu contează ordinea în care sunt considerate punctele, nici ordinea în care sunt trecute coordonatele segmentelor în fișierul de ieșire.
11) Dacă prin calcule matematice rezultă numere reale pentru coordonate, acestea vor fi ajustate la cele mai apropiate numere naturale (plus sau minus o unitate).
12) Programul va fi realizat în limbajul C/C++/Pascal. Dacă se folosește Pascal, va trebui să apelați la o bibliotecă grafică similară (și să ne trimiteți e-mail cu precizarea acestui fapt).
13) Pentru orice alte nelămuriri, contactați-l pe domnul Bogdan Pătruț, pe serverul de Discord al disciplinei „Introducere în programare”.

Exemplu:
pentru șirul de intrare V(V,A,V) se poate obține fișierul de ieșire TRIUNGHI.TXT cu următorul conținut și reprezentarea grafică alăturată.

250 214 550 214 250 214 400 472 550 214 400 472 475 171 625 171 475 171 550 300 625 171 550 300 325 515 475 515 325 515 400 386 475 515 400 386 175 171 325 171 175 171 250 300 325 171 250 300	

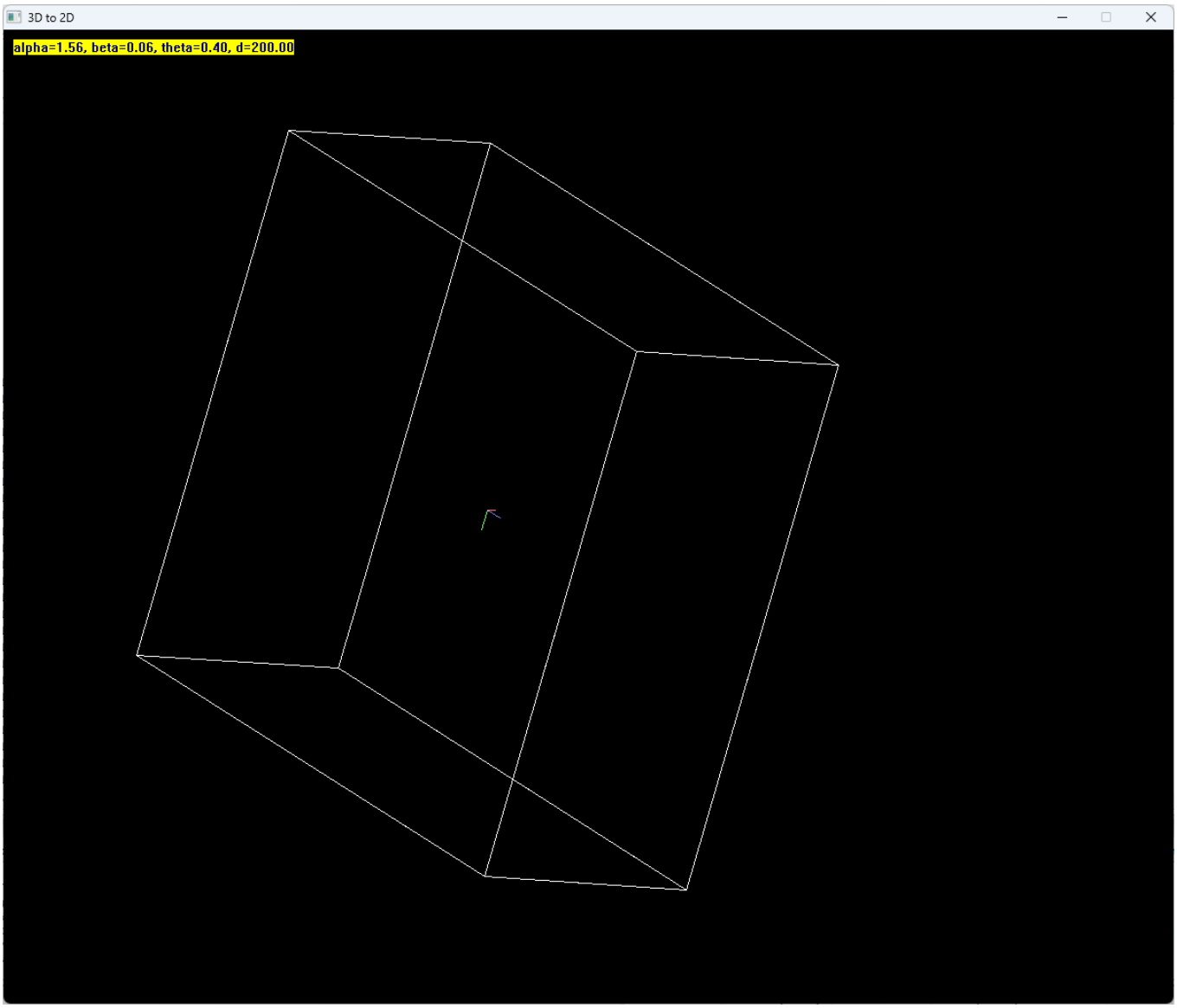
Teste
A 1
A(V,V,A) 1 5 2 4 2 4 2 6
V(#,A,V) etc.
V(A(V,V,V),V(A,A,A),V(V,A,V))
V(#,#,A(A(#,V,V),#,V)), A(#,#,A(#,#,A(#,A))), V(A(V(A(V(A)))))), A(A(A(#,#,V),#,V(A(V,V),#,V)))

10.5. Editor de expresii

10.6. Arbori Quad și vizualizator de imagini

Clasa a XII-a

Editor 3D




```
#include <cmath>
#include <winbgim.h>

using namespace std;

struct point3D
{
    double x, y, z;
};

struct camera
{
    double d, alpha, beta, theta;
};

struct point2D
{
    double X, Y;
};
```

```
void transformPoint(point3D P, camera C, point2D &Q)
{
    // Translate point to camera origin
    double x1 = P.x;
    double y1 = P.y;
    double z1 = P.z - C.d;

    // Rotate around y-axis by beta angle
    double cosb = cos(C.beta);
    double sinb = sin(C.beta);
    double x2 = x1 * cosb + z1 * sinb;
    double y2 = y1;
    double z2 = -x1 * sinb + z1 * cosb;

    // Rotate around z-axis by theta angle
    double cost = cos(C.theta);
    double sint = sin(C.theta);
    double x3 = x2 * cost - y2 * sint;
    double y3 = x2 * sint + y2 * cost;
    double z3 = z2;

    // Project onto 2D plane
    Q.X = 600+z3 / C.d * tan(C.alpha) + x3;
    Q.Y = 500+y3 / C.d * tan(C.alpha) + y3;
}


int main()
{
    point3D P[20];
    point2D Q[20];
    camera C;
    C.alpha=1.56;
    C.beta=0.30;
    C.theta=0.40;
    C.d=200;
    char ss[200];
    P[1].x=-200;
    P[1].y=-200;
    P[1].z=-200;
    P[2].x=200;
    P[2].y=-200;
    P[2].z=-200;
    P[3].x=200;
    P[3].y=200;
    P[3].z=-200;
    P[4].x=-200;
    P[4].y=200;
    P[4].z=-200;
    P[5].x=-200;
    P[5].y=-200;
    P[5].z=200;
    P[6].x=200;
```

```
    P[6].y=-200;
    P[6].z=200;
    P[7].x=200;
    P[7].y=200;
    P[7].z=200;
    P[8].x=-200;
    P[8].y=200;
    P[8].z=200;
    P[9].x=0;
    P[9].y=0;
    P[9].z=0;
    P[10].x=15;
    P[10].y=0;
    P[10].z=0;
    P[11].x=0;
    P[11].y=15;
    P[11].z=0;
    P[12].x=0;
    P[12].y=0;
    P[12].z=15;
    initwindow(1200,1000,"3D to 2D");
    int i; char tasta;
    setbkcolor(YELLOW);
    setcolor(BLUE);
    sprintf(ss,"alpha=%0.2f, beta=%0.2f, theta=%0.2f, d=%0.2f",
C.alpha,C.beta,C.theta,C.d);
    outtextxy(10,10,ss);
    setbkcolor(BLACK);
    int pagina=1;
    setactivepage(pagina); setvisualpage(1-pagina);
    setbkcolor(YELLOW);
    setcolor(BLUE);
    sprintf(ss,"alpha=%0.2f, beta=%0.2f, theta=%0.2f, d=%0.2f",
C.alpha,C.beta,C.theta,C.d);
    outtextxy(10,10,ss);
    setbkcolor(BLACK);
    do
    {
        for (i=1; i<=12; i++)
        {
            transformPoint(P[i],C,Q[i]);
        }
        setcolor(WHITE);
        line(Q[1].X,Q[1].Y,Q[2].X,Q[2].Y);
        line(Q[2].X,Q[2].Y,Q[3].X,Q[3].Y);
        line(Q[3].X,Q[3].Y,Q[4].X,Q[4].Y);
        line(Q[4].X,Q[4].Y,Q[1].X,Q[1].Y);
        line(Q[5].X,Q[5].Y,Q[6].X,Q[6].Y);
        line(Q[6].X,Q[6].Y,Q[7].X,Q[7].Y);
        line(Q[7].X,Q[7].Y,Q[8].X,Q[8].Y);
        line(Q[8].X,Q[8].Y,Q[5].X,Q[5].Y);
        line(Q[1].X,Q[1].Y,Q[5].X,Q[5].Y);
        line(Q[2].X,Q[2].Y,Q[6].X,Q[6].Y);
```

```
        line(Q[3].X,Q[3].Y,Q[7].X,Q[7].Y);
        line(Q[4].X,Q[4].Y,Q[8].X,Q[8].Y);
        setcolor(LIGHTBLUE);
        line(Q[9].X,Q[9].Y,Q[10].X,Q[10].Y);
        setcolor(LIGHTGREEN);
        line(Q[9].X,Q[9].Y,Q[11].X,Q[11].Y);
        setcolor(LIGHTRED);
        line(Q[9].X,Q[9].Y,Q[12].X,Q[12].Y);
        //delay(500);
        pagina=1-pagina;
        setactivepage(pagina); setvisualpage(1-pagina);
        tasta=getch();
        if (tasta==0)
        {
            tasta=getch();
            if (tasta==72) C.theta+=0.01;
            if (tasta==80) C.theta-=-0.01;
            if (tasta==75) C.beta+=0.01;
            if (tasta==77) C.beta-=0.01;
        }
        else
        {
            if (tasta=='z') C.alpha+=0.01;
            if (tasta=='x') C.alpha-=0.01;
            if (tasta=='a') C.d=C.d+20;
            if (tasta=='q') C.d=C.d-100;
        }
        cleardevice();
        setbkcolor(YELLOW);
        setcolor(BLUE);
        sprintf(ss,"alpha=%0.2f, beta=%0.2f, theta=%0.2f, d=%0.2f",
                C.alpha,C.beta,C.theta,C.d);
        outtextxy(10,10,ss);
        setbkcolor(BLACK);
    }
    while (tasta!=27);
    closegraph();
    return 0;
}
```

Etape în dezvoltarea jocului cu Mario

1) Mario se misca stanga dreapta, pe loc (fara sa dea din picioare, deci fara alte ipostaze)

```
#include <winbgim.h>

#define STG 75
#define DRP 77
#define ESC 27

int directie;

void deseneazaMario()
{
    if (directie==DRP)
        readimagefile("mario1.jpg",100,100,200,220);
    if (directie==STG)
        readimagefile("mario2.jpg",100,100,200,220);
}

int main()
{
    initwindow(300,300,"Demo");
    outtextxy(10,10,"Apasa sagetile stanga/dreapta");
    directie=DRP;
    deseneazaMario();
    char tasta;
    do {
        tasta=getch(); if (tasta==0) tasta=getch();
        directie=tasta;
        deseneazaMario();
    } while (tasta!=ESC);
```

```
    closegraph();
    return 0;
}
```

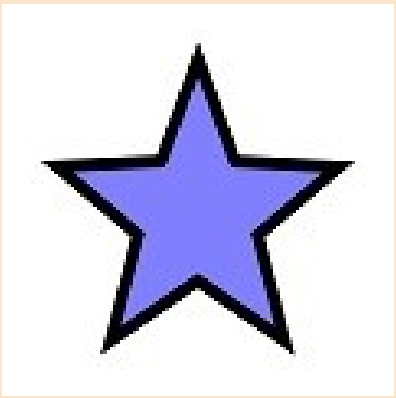
2) Se incarca o harta cu un labirint, dintr-un fisier text
Fisierul "harta.txt":

```
8 10
@@@@@@@@@@
@..*..@.*@
@....@@..@
@.@@.M..@@
@*....@@@@
@@@...@.*@
@.*......@
@@@@@@@@@@
```

```
#include <iostream>
#include <fstream>
#include <winbgim.h>

#define LAT 100
#define MAX 20

using namespace std;

int nrLinii,nrColoane;
int imario,jmario,nrStelute;

char harta[MAX][MAX];

void citesteHarta()
{
    ifstream fisier("harta.txt");
    fisier>>nrLinii>>nrColoane;
    int i,j; char car;
    nrStelute=0;
    for (i=0;i<nrLinii;i++)
        for (j=0;j<nrColoane;j++)
        {
            fisier>>car;
            if (car=='M') {imario=i;jmario=j; car='.';}
            if (car=='*') nrStelute++;
            harta[i][j]=car;
        }
    fisier.close();
}

void deseneazaHarta()
{
    int x,y,i,j;
    for (i=0;i<nrLinii;i++)
        for (j=0;j<nrColoane;j++)
        {
            x=LAT*j; y=LAT*i;
            if (harta[i][j]=='@')
```

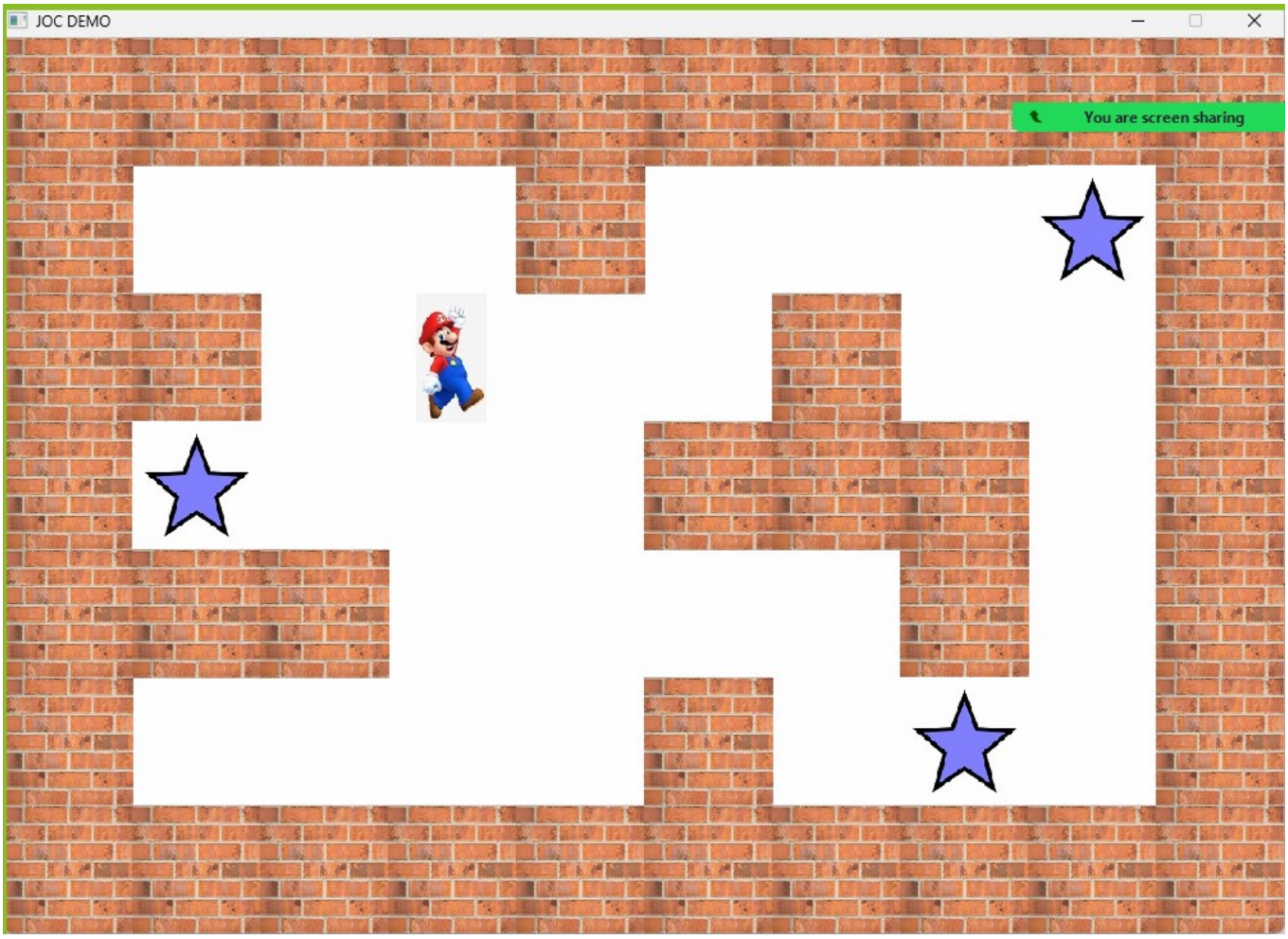
```
                readimagefile("caramida.jpg",x,y,x+LAT,y+LAT);
            if (harta[i][j]=='.')
                readimagefile("spatiu.jpg",x,y,x+LAT,y+LAT);
            if (harta[i][j]=='*')
                readimagefile("stea.jpg",x,y,x+LAT,y+LAT);
        }
}

int main()
{
    citesteHarta();
    initwindow(LAT*nrColoane,LAT*nrLinii,"Demo");
    deseneazaHarta();
    getch();
    closegraph();
    cout<<"Mario este la linia "<<imario<<" si coloana "<<jmario<<"."<<endl;
    cout<<"Sunt "<<nrStelute<<" stelute.";
    return 0;
}
```

Fisierele cu imagini

caramida.jpg	spatiu.jpg	 stea.jpg

```
#include <iostream>
#include <fstream>
#include <winbgim.h>

#define MAX 20
#define LAT 100

using namespace std;

char lab[MAX][MAX];

int nrLinii, nrColoane, imario, jmario, nrStelute;

void incarcaLabirint()
{
    char car; int i,j;
    ifstream fisier("lab.txt");
    fisier>>nrLinii>>nrColoane;
    for (i=0;i<nrLinii;i++)
        for (j=0;j<nrColoane;j++)
            {
                fisier>>car;
                if (car=='M')
                    { imario=i; jmario=j; car='.'; }
                if (car=='*')
                    nrStelute++;
                lab[i][j]=car;
            }
    fisier.close();
}

void deseneazaLabirint()
{
    int i,j;
    for (i=0;i<nrLinii;i++)
        for (j=0;j<nrColoane;j++)
            {
                if (lab[i][j]=='*')
                    readimagefile("stea.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
                if (lab[i][j]=='@')
                    readimagefile("caramida.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
            }
}

void deseneazaMario()
{
    readimagefile("mario1.jpg",LAT*jmario,LAT*imario,LAT*(jmario+1),LAT*(imario+1));
}

int main()
{
    incarcaLabirint();
    //cout<<imario<<" "<<jmario<<" "<<nrStelute;
    initwindow(LAT*nrColoane,LAT*nrLinii,"JOC DEMO");
    setbkcolor(WHITE); cleardevice();
```

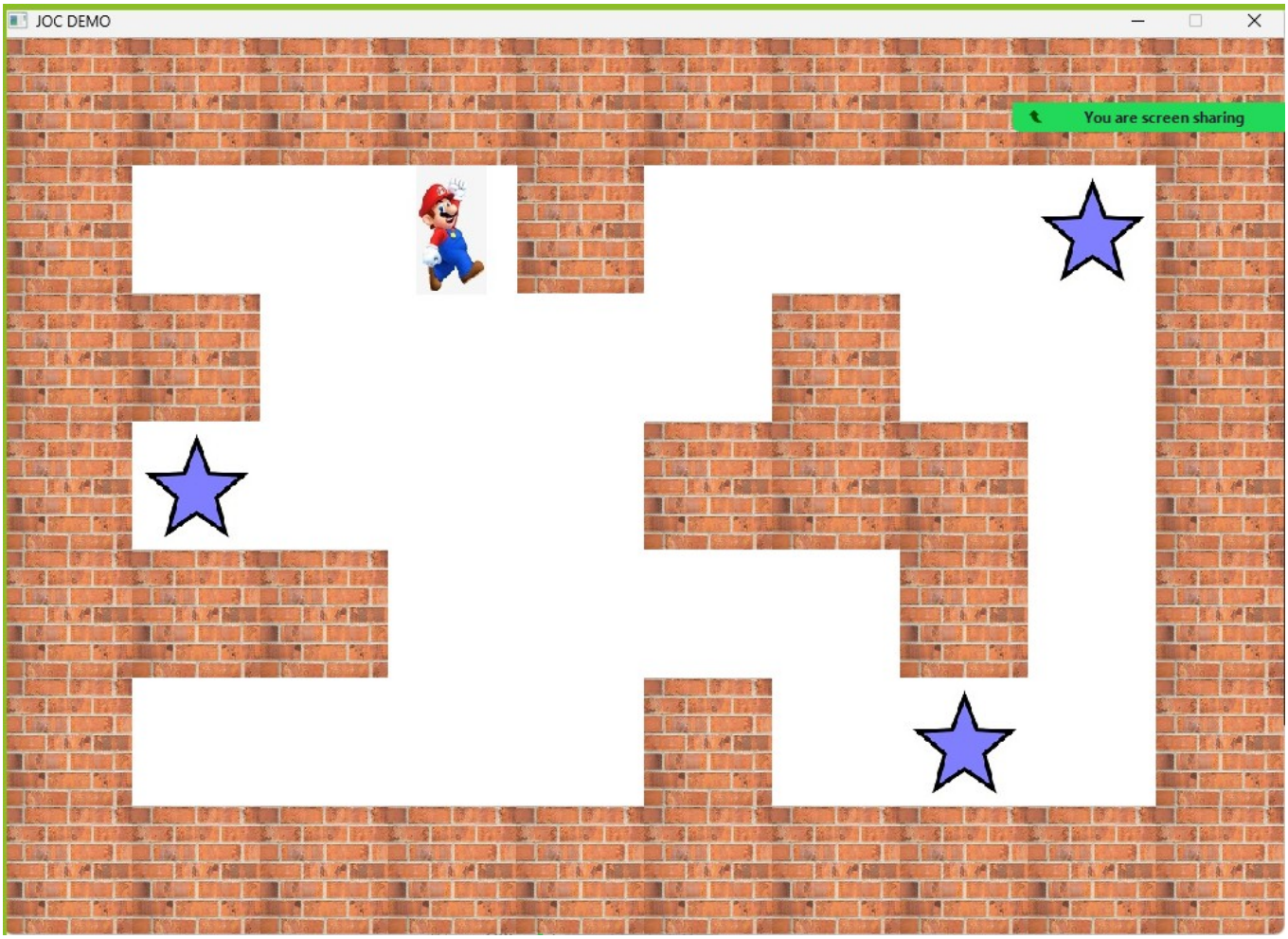
```
    deseneazaLabirint();
    deseneazaMario();
    getch();
    closegraph();
    return 0;
}
```




```
#include <iostream>
#include <fstream>
#include <winbgim.h>

#define MAX 20
#define LAT 100
#define ESC 27
#define SUS 72
#define STG 75
#define DRP 77
#define JOS 80

using namespace std;

char lab[MAX][MAX];

int nrLinii, nrColoane, imario, jmario, nrStelute, scor;

void incarcaLabirint()
{
```

```
    char car;
    int i,j;
    ifstream fisier("lab.txt");
    fisier>>nrLinii>>nrColoane;
    for (i=0; i<nrLinii; i++)
        for (j=0; j<nrColoane; j++)
        {
            fisier>>car;
            if (car=='M')
            {
                imario=i;
                jmario=j;
                car='.';
            }
            if (car=='*')
                nrStelute++;
            lab[i][j]=car;
        }
    fisier.close();
}

void deseneazaLabirint()
{
    int i,j;
    for (i=0; i<nrLinii; i++)
        for (j=0; j<nrColoane; j++)
        {
            if (lab[i][j]=='*')
                readimagefile("stea.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
            if (lab[i][j]=='@')
                readimagefile("caramida.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
            if (lab[i][j]=='.')
                readimagefile("spatiu.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
        }
}

void deseneazaMario()
{
    readimagefile("mario1.jpg",LAT*jmario,LAT*imario,LAT*(jmario+1),LAT*(imario+1));
}

void stergeMario()
{
    readimagefile("spatiu.jpg",LAT*jmario,LAT*imario,LAT*(jmario+1),LAT*(imario+1));
}

int main()
{
    incarcaLabirint();
    //cout<<imario<<" "<<jmario<<" "<<nrStelute;
    initwindow(LAT*nrColoane,LAT*nrLinii,"JOC DEMO");
    //setbkcolor(WHITE); cleardevice();
    deseneazaLabirint();
    deseneazaMario();
    scor=0;
```

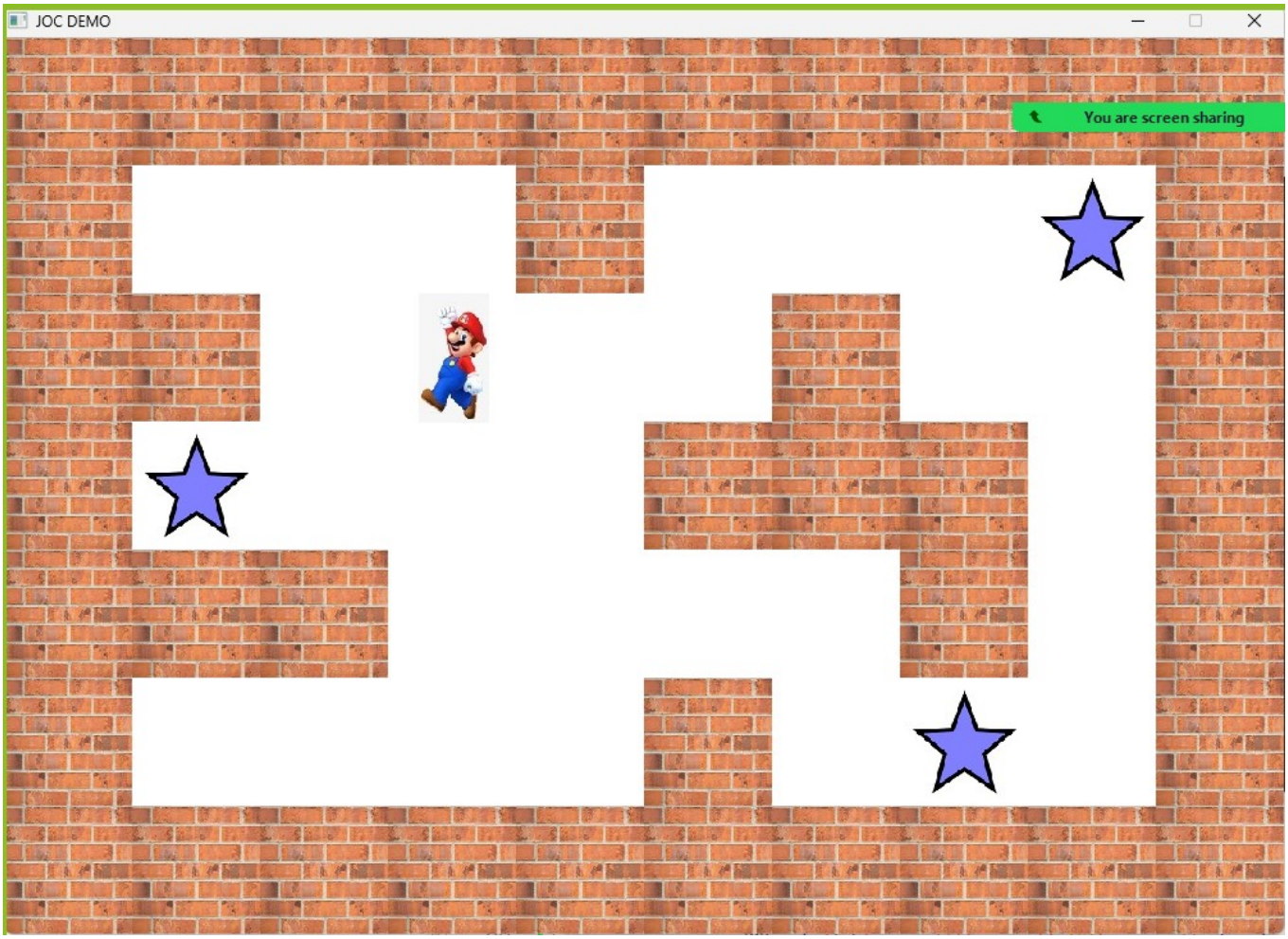
```
    char tasta;
    do
    {
            tasta=getch();
            if (tasta==0)
            {
                tasta=getch();
                stergeMario();
                switch(tasta)
                {
                case SUS:
                    if (lab[imario-1][jmario]!='@') imario--;
                    break;
                case JOS:
                    if (lab[imario+1][jmario]!='@') imario++;
                    break;
                case STG:
                    if (lab[imario][jmario-1]!='@') jmario--;
                    break;
                case DRP:
                    if (lab[imario][jmario+1]!='@') jmario++;
                    break;
                }
                if (lab[imario][jmario]=='*')
                {
                    scor++;
                    lab[imario][jmario]='.';
                }
                deseneazaMario();
            }
    }
    while (tasta!=ESC && scor!=nrStelute);
    //getch();
    closegraph();
    return 0;
}
```




```
#include <iostream>
#include <fstream>
#include <winbgim.h>
#include <windows.h>

#define MAX 20
#define LAT 100
#define ESC 27
#define SUS 72
#define STG 75
#define DRP 77
#define JOS 80

using namespace std;

char lab[MAX][MAX];

int nrLinii, nrColoane, imario, jmario, nrStelute, scor;

int directie;

void incarcaLabirint()
{
    char car;
```

```
    int i,j;
    ifstream fisier("lab.txt");
    fisier>>nrLinii>>nrColoane;
    for (i=0; i<nrLinii; i++)
        for (j=0; j<nrColoane; j++)
        {
            fisier>>car;
            if (car=='M')
            {
                imario=i;
                jmario=j;
                car='.';
            }
            if (car=='*')
                nrStelute++;
            lab[i][j]=car;
        }
    fisier.close();
}

void deseneazaLabirint()
{
    int i,j;
    for (i=0; i<nrLinii; i++)
        for (j=0; j<nrColoane; j++)
        {
            if (lab[i][j]=='*')
                readimagefile("stea.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
            if (lab[i][j]=='@')
                readimagefile("caramida.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
            if (lab[i][j]=='.')
                readimagefile("spatiu.jpg",LAT*j,LAT*i,LAT*(j+1),LAT*(i+1));
        }
}

void deseneazaMario()
{
    if (directie==DRP)

readimagefile("mario1.jpg",LAT*jmario,LAT*imario,LAT*(jmario+1),LAT*(imario+1));
    if (directie!=DRP)

readimagefile("mario2.jpg",LAT*jmario,LAT*imario,LAT*(jmario+1),LAT*(imario+1));
}

void stergeMario()
{
    readimagefile("spatiu.jpg",LAT*jmario,LAT*imario,LAT*(jmario+1),LAT*(imario+1));
}

int main()
{
    incarcaLabirint();
    //cout<<imario<<" "<<jmario<<" "<<nrStelute;
    initwindow(LAT*nrColoane,LAT*nrLinii,"JOC DEMO");
```

```
    //setbkcolor(WHITE); cleardevice();
    deseneazaLabirint();
    PlaySound("SuperMarioBros.wav",NULL,SND_ASYNC);
    deseneazaMario();
    scor=0;
    char tasta;
    directie=0;

    do
    {
        //stergeMario();
       // if (kbhit())
        {
            tasta=getch();
            if (tasta==0)
            {
                tasta=getch();
                directie=tasta;
                stergeMario();
            }
        }
        switch(directie) {
            case SUS: if (lab[imario-1][jmario]!='@') imario--; break;
            case JOS: if (lab[imario+1][jmario]!='@') imario++; break;
            case STG: if (lab[imario][jmario-1]!='@') jmario--; break;
            case DRP: if (lab[imario][jmario+1]!='@') jmario++; break;
        }
        deseneazaMario();
        if (lab[imario][jmario]=='*')
        {
            scor++; lab[imario][jmario]='.';
            Beep(1000,70);Beep(1200,70);Beep(800,70);
        }
        delay(200);
    }
    while (tasta!=ESC && scor!=nrStelute);
    //getch();
    closegraph();
    return 0;
}
```


JOC DEMO
You are screen sharing

POKER

```
// POKER - Bogdan Patrut, 9.03.2023
//
// Carti de joc disponibile la
// https://www.improvemagic.com/all-playing-cards-names-with-pictures/?
utm_content=cmp-true
/*
rezulta o configuratie

0 - nimic: 9 10 Q K A (nu toate de aceeasi culoare)
1 - o pereche: J J
2 - doua perechi: J J Q Q 10
3 - trei bucati: J J J Q A
4 - chinta mica: 9 10 J Q K (sa nu fie toate de aceeasi culoare)
5 - chinta mare: 10 J Q K A (sa nu fie toate de aceeasi culoare)
6 - mana de culoare: 5 carti oarecare de aceeasi culoare
7 - full: 10 10 10 Q Q (ful de decari cu dame in coada)
```

```
8 - carre: K K K K 9 (caree de popi cu 9 in coada)
9 - chinta roiala: chinta cu aceeasi culoare
*/

#include <iostream>
#include <string>
#include <winbgim.h>
#include <time.h>

#define FRACTIE 0.5
#define LAT 224
#define HAS 312
#define LIMITA 9

// variante simplificate la configuratii
#define CHINTA_ROIALA 10
#define CHINTA_DE_CULOARE 9
#define CARRE 8
#define FULL 7
#define MANA_CULOARE 6
#define CHINTA 5
#define TREI_BUCATI 4
#define DOUA_PERECHI 3
#define O_PERECHE 2
#define CARTEA_MAXIMA 1
#define NIMIC 0

using namespace std;

struct carte
{
    char simbol;  // 'l'=romb (caro), 's'=cupa (inima rosie), 'k' = trefla, 'p'=pica
(frunza)
    int numar; // numar: 1=A (as), 2, 3, ....,10, 11 = J, 12 = Q, 13 = K
    bool stare; // true = cu fata in sus, false = cu fata in jos
};

carte pachet[52]; // de la 0 la 51
carte pOm[5];
carte pCalc[5];
int pointerCarti;

void poza(carte c, int x, int y, float fractie, bool tip)
// incarca imaginea din fisierul JPG corespunzator unei carti de joc c
// coltul din stanga sus va fi in punctul de coordonate x si y
{
    string fisier;
    switch(c.numar)
    {
    case 1:
        fisier="a";
        break; // as 1 ->"a"
    case 11:
        fisier="j";
        break; // valet 11 -> "j"
```

```
    case 12:
        fisier="q";
        break; // dama
    case 13:
        fisier="k";
        break; // rege
    default:
        fisier=to_string(c.numar);  // pt. numerele de la 2 la 10, de exemplu 7 ->
"7"
    }
    fisier=c.simbol+fisier+".jpg";
    //cout<<fisier<<endl;
    if (tip)
        readimagefile(fisier.c_str(),x,y,x+LAT*fractie,y+HAS*fractie);
    else
        readimagefile("spate.jpg",x,y,x+LAT*fractie,y+HAS*fractie);
}

void generareCarti()
// 'k' 1 (as de trefla), 'k' 2 (2-ul de trefla) .... 'k' 13 (cartile de la 0 la 12)
// 'l' 1 (as de romb), 'l' 2 (2-ul de romb) ..... 'l' 13 (cartile de la 13 la 25)
// 'p' 1 (a de frunza), 'p' 2 (2-ul de frunza) samd
// 's' 1 (as de inima) .... (ultimele 13 carti)
{
    int i,j;
    char sir[5]="klps";
    for (i=0; i<13; i++)
    {
        for (j=0; j<5; j++)
        {
            pachet[4*i+j].numar=i+1;
            pachet[4*i+j].simbol=sir[j];
            pachet[4*i+j].stare=false;
        }
    }
    for (i=0; i<4; i++)
        pachet[i].numar=14; // Asii devin 14
}

void amestecareCarti(int vmin)
{
    int sarite=(vmin-1)*4;
    int i,c1,c2;
    for (i=1; i<=200; i++)
    {
        c1=sarite+rand()%(52-sarite);
        c2=sarite+rand()%(52-sarite);
        swap(pachet[c1],pachet[c2]);
    }
}

void afisareCarti()
{
    int i,j,k=0;
```

```
    for (i=0; i<13; i++)
        for (j=0; j<4; j++)
        {
            poza(pachet[k],LAT*FRACTIE*i,HAS*FRACTIE*j,FRACTIE,pachet[k].stare);
            delay(10);
            k++;
        }
}

void impartireCarti()
{
    int i;
    // primul jucator, omul
    for (i=0; i<5; i++)
    {
        pOm[i]=pachet[4*(LIMITA-1)+i];
        pOm[i].stare=true;
    }
    // al doilea jucator, calculatorul
    for (i=5; i<10; i++)
        pCalc[i-5]=pachet[4*(LIMITA-1)+i];
    pointerCarti=4*(LIMITA-1)+10;
}

void afiseazaCartiOm()
{
    for (int i=0; i<5; i++)
        poza(pOm[i],(4+i)*LAT*FRACTIE,HAS*3*FRACTIE,FRACTIE,pOm[i].stare);
}

void afiseazaCartiCalc()
{
    for (int i=0; i<5; i++)
        poza(pCalc[i],(4+i)*LAT*FRACTIE,0,FRACTIE,pCalc[i].stare);
}

void sortareCarti(carte p[5])
{
    for (int i=0; i<4; i++)
        for (int j=i+1; j<5; j++)
            if (p[j].numar<p[i].numar)// && p[j].numar!=1)
                swap(p[i],p[j]);
}

void afiseazaButon(char* text, int x, int y)
{
    setfillstyle(SOLID_FILL,YELLOW);
    setbkcolor(YELLOW);
    setcolor(LIGHTRED);
    bar(x,y+100,x+LAT*FRACTIE,y+HAS*FRACTIE-100);
    outtextxy(x,y+HAS/2*FRACTIE-5,text);
}

bool apartine(int x, int y, int x1, int y1, int x2, int y2)
{
```

```
    return x1<=x && x<=x2 && y1<=y && y<=y2;
}

int numarCartiOmIntoarse()
{
    int numar=0;
    for (int i=0; i<5; i++)
        if (pOm[i].stare==false)
            numar++;
    return numar;
}

void schimbaCartiOm()
{
    int i,x,y,x1,y1,x2,y2;
    int gasit=0;
    bool gata=false;
    do
    {
        if (ismouseclick(WM_LBUTTONDOWN))
        {
            clearmouseclick(WM_LBUTTONDOWN);
            x=mousex();
            y=mousey();
            for (i=0; i<7; i++)
            {
                if (i==5) continue;
                x1=LAT*(4+i)*FRACTIE;
                y1=HAS*3*FRACTIE;
                x2=x1+FRACTIE*LAT;
                y1=FRACTIE*HAS*3;
                y2=y1+FRACTIE*HAS;
                //if (i==1) cout<<x<<" "<<y<<" "<<x1<<" "<<y1<<" "<<x2<<"
"<<y2<<endl;
                if (apartine(x,y,x1,y1,x2,y2))
                {
                    gasit=i;
                    //cout<<i<<" ";
                    if (i<5 && pOm[i].stare==false || pOm[i].stare==true &&
numarCartiOmIntoarse()<3)
                    {
                        pOm[i].stare=!pOm[i].stare;
                        afiseazaCartiOm();
                    }
                    else
                        gata=true;
                }
            }
        }
    }
    while (!gata);
    for (i=0; i<5; i++)
        if (pOm[i].stare==false)
        {
            pOm[i]=pachet[pointerCarti];
```

```
            pOm[i].stare=true;
            pointerCarti++;
        }
}

void schimbaCartiCalc()
{
    int n=rand()%4;
    char text[30];
    if (n==0) strcpy(text,"PAS.");
    else sprintf(text,"Eu am schimbat %d carti.",n);
    setbkcolor(YELLOW);
    setcolor(LIGHTRED);
    outtextxy(LAT*FRACTIE*2+40,100,text);
    delay(2000);
    readimagefile("femeie.jpg",0,0,4*FRACTIE*LAT,4*FRACTIE*HAS);
    //setbkcolor(BLACK); setcolor(BLACK);
    //outtextxy(LAT*FRACTIE*3,HAS*FRACTIE+10,text);
}
/*
int putereConfiguratie(carte p[5])
{
    int i;
    if (p[0].simbol==p[1].simbol && p[1].simbol==p[2].simbol &&
p[2].simbol==p[3].simbol &&
        p[3].simbol==p[4].simbol) return CHINTA_ROIALA;
    int fr[5]={0,0,0,0,0};
    for (i=0;i<5;i++)
        fr[p[i].numar]++;
    for (i=0;i<5;i++)
        if (fr[i]==4) return CARRE;

    string config="";

    for (i=0;i<4;i++)
    {
        switch(p[i].numar) {
            case 10: config+="X"; break;
            case 11: config+="J"; break;
            case 12: config+="Q"; break;
            case 13: config+="K"; break;
            case 14: config+="A"; break;
            default: config+=to_string(p[i].numar);
        }
    }

    if (config=="XJQKA") return CHINTA_MARE;
    if (config=="9XJQK") return CHINTA_MICA;
}
*/

int putereMana(carte mana[5])
{
    // mana este sortata dupa numar
    /*    for (int i = 0; i < 4; i++)
```

```
    {
        for (int j = i+1; j < 5; j++)
        {
            if (mana[j].numar < mana[i].numar)
            {
                carte temp = mana[i];
                mana[i] = mana[j];
                mana[j] = temp;
            }
        }
    }
*/
// Check for a straight or straight flush
bool straight = true;
bool straight_flush = true;
int straight_high_card = mana[4].numar;
for (int i = 1; i < 5; i++)
{
    if (mana[i].numar != mana[i-1].numar + 1)
    {
        straight = false;
    }
    if (mana[i].simbol != mana[i-1].simbol)
    {
        straight_flush = false;
    }
    if (mana[i].numar != mana[i-1].numar + 1 && i < 4)
    {
        straight_high_card = mana[i-1].numar;
    }
}

// Check for a flush
bool flush = true;
for (int i = 1; i < 5; i++)
{
    if (mana[i].simbol != mana[0].simbol)
    {
        flush = false;
        break;
    }
}
// Check for pairs, three of a kind, and four of a kind
int pairs = 0;
int triple = 0;
int four_of_a_kind = 0;
int high_card = mana[4].numar;
for (int i = 0; i < 3; i++)
{
    if (mana[i].numar == mana[i+1].numar && mana[i+1].numar!=mana[i+2].numar)
    {
        pairs++;
        if (mana[i].numar > high_card)
        {
            high_card = mana[i].numar;
```

```
            }
        }
        if (mana[i].numar == mana[i+1].numar && mana[i+1].numar == mana[i+2].numar)
        {
            triple = mana[i].numar;
        }
        if (mana[i].numar == mana[i+1].numar && mana[i+1].numar == mana[i+2].numar
&& mana[i+2].numar == mana[i+3].numar)
        {
            four_of_a_kind = mana[i].numar;
        }
    }
    if (mana[3].numar == mana[4].numar && mana[2].numar == mana[3].numar &&
mana[1].numar == mana[2].numar)
    {
        pairs++;
        if (mana[3].numar > high_card)
        {
            high_card = mana[3].numar;
        }
    }
    if (mana[2].numar == mana[3].numar && mana[3].numar == mana[4].numar)
    {
        triple = mana[2].numar;
    }
    if (mana[1].numar == mana[2].numar && mana[2].numar == mana[3].numar &&
mana[3].numar == mana[0].numar)
    {
        four_of_a_kind = mana[1].numar;
    }
    if (mana[1].numar == mana[2].numar && mana[2].numar == mana[3].numar &&
mana[3].numar == mana[4].numar)
    {
        four_of_a_kind = mana[1].numar;
    }
    // Determine the hand rank
    if (straight_flush && straight_high_card == 14)
    {
        return CHINTA_ROIALA;
    }
    else if (straight_flush)
    {
        return CHINTA_DE_CULOARE;
    }
    else if (four_of_a_kind)
    {
        return CARRE;
    }
    else if (triple && pairs==2)
    {
        return FULL;
    }
    else if (flush)
    {
        return MANA_CULOARE;
```

```
    }
    else if (straight)
    {
        return CHINTA;
    }
    else if (triple)
    {
        return TREI_BUCATI;
    }
    else if (pairs == 2)
    {
        return DOUA_PERECHI;
    }
    else if (pairs == 1)
    {
        return O_PERECHE;
    }
    else
    {
        return CARTEA_MAXIMA;
    }
    return NIMIC;
}

int main()
{
    srand(time(NULL));
    initwindow(LAT*FRACTIE*13,HAS*FRACTIE*4,"POKER",200,200);
    readimagefile("femeie.jpg",0,0,4*FRACTIE*LAT,4*FRACTIE*HAS);
    readimagefile("barbat.jpg",9*FRACTIE*LAT,0,13*FRACTIE*LAT,4*FRACTIE*HAS);
    readimagefile("dealer.jpg",4*FRACTIE*LAT,0,9*FRACTIE*LAT,4*FRACTIE*HAS);
    generareCarti();
    //setvisualpage(0);
    //setactivepage(1);
    //afisareCarti();
    //setvisualpage(1);
    //getch();
    amestecareCarti(LIMITA);
    //afisareCarti();
    impartireCarti();
    //afiseazaCartiOm(true);
    sortareCarti(pOm);
    sortareCarti(pCalc);
    afiseazaCartiOm();
    //afiseazaCartiCalc();
    afiseazaButon("SCHIMBA CARTI",FRACTIE*LAT*10,3*FRACTIE*HAS);
    schimbaCartiOm();
    sortareCarti(pOm);
    afiseazaCartiOm();
    schimbaCartiCalc();
    for (int i=0; i<5; i++)
        pCalc[i].stare=true;
    afiseazaCartiCalc();
    if (putereMana(pOm)>putereMana(pCalc))
        outtextxy(5*FRACTIE*LAT,2*FRACTIE*HAS,"Felicitari! Ai castigat.");
```

```
    else
        if (putereMana(pOm)<putereMana(pCalc))
            outtextxy(5*FRACTIE*LAT,2*FRACTIE*HAS,"Imi pare rau! Ai pierdut.");
        else
            outtextxy(5*FRACTIE*LAT,2*FRACTIE*HAS,"Egalitate! Ambele maini au
aceeasi putere.");
    cout<<"Om: "<<putereMana(pOm)<<endl;
    cout<<"Calc: "<<putereMana(pCalc)<<endl;
    getch();
    closegraph();
    return 0;
}
```

.*tuse.*febra.*=>Mergi la un doctor!|Ia paracetamol si bea ceaiuri.|Mergi la farmacie si intreaba de medicamentele potrivite.
.*nu.*bani.*=>Ia-ti un job.|Mergi la o banca si imprumuta bani.|Vorbeste cu seful pentru o crestere de salariu
.*doctorul.*nu.*ma.*intelege.*=>Ce te face sa crezi asta?|Poti sa mergi la un alt medic.|Consulta un tratat de medicina.
.*racit.*=>Cum se manifesta raceala?|Cine ti-a pus diagnosticul?|Poate e vorba despre o alergie.
.*am temperatura.*=>Vrei sa spui ca ai febra?|Tusesti?
.*tuse.*=>Posibil sa fie gripa, raceala sau o alergie.|Un medic ORL ar trebui sa te consulte.
.*antihistaminice.*=>Da, antihistaminicele sunt bune pentru alergii.|Daca ai vreo infectie bacteriana, va trebui sa iei antibiotice.
.*antibiotice.*=>Medicul poate prescrie antibiotice.|Daca ai o viroza respiratorie, antibioticele nu te ajuta.|Consulta un medic.
.*(oboseala|dureri musculare|febra).*=>TRA LA LA

.*tuse.*febra.*=>Mergi la un doctor!|Ia paracetamol si bea ceaiuri.|Mergi la farmacie si intreaba de medicamentele potrivite.
.*nu.*bani.*=>Ia-ti un job.|Mergi la o banca si imprumuta bani.|Vorbeste cu seful pentru o crestere de salariu
.*doctorul.*nu.*ma.*intelege.*=>Ce te face sa crezi asta?|Poti sa mergi la un alt medic.|Consulta un tratat de medicina.
.*racit.*=>Cum se manifesta raceala?|Cine ti-a pus diagnosticul?|Poate e vorba despre o alergie.
.*am temperatura.*=>Vrei sa spui ca ai febra?|Tusesti?
.*tuse.*=>Posibil sa fie gripa, raceala sau o alergie.|Un medic ORL ar trebui sa te consulte.
.*antihistaminice.*=>Da, antihistaminicele sunt bune pentru alergii.|Daca ai vreo infectie bacteriana, va trebui sa iei antibiotice.
.*antibiotice.*=>Medicul poate prescrie antibiotice.|Daca ai o viroza respiratorie, antibioticele nu te ajuta.|Consulta un medic.
.*(oboseala|dureri musculare|febra).*=>TRA LA LA
.*senzatie de arsura in piept)|(?=.*greata)|(?:senzatie de arsura in piept|greata)+$ => Ai o afectiune gastro-esofagiana, evita alimentele acide, poti lua antiacide si sa consulti un medic.
^(?=.*tuse)|(?:tuse)+$ => Ai o afectiune respiratorie, ia sirop de tuse si fa aburi cu apa fierbinte.
^(?=.*durere de cap)|(?:durere de cap)+$ => Ai o durere de cap, poti lua analgezice si sa te odihnesti.
^(?=.*dificultati de respiratie)|(?=.*durere toracica)|(?:dificultati de respiratie|durere toracica)+$ => Ai o afectiune respiratorie sau cardiaca, mergi la medic imediat.
^(?=.*febra)|(?:febra)+$ => Ai o infectie, ia antipiretice si fa odihna.
^(?=.*dureri de burta)|(?=.*balonare)|(?:dureri de burta|balonare)+$ => Ai o afectiune gastro-intestinala, evita alimentele grele si grase, poti lua medicamente antispastice si sa consulti un medic.
^(?=.*senzatie de furnicaturi in brate)|(?:senzatie de furnicaturi in brate)+$ => Ai o afectiune neurologica sau circulatorie, mergi la medic imediat.
^(?=.*tuse)(?=.*febra)|(?=.*dureri musculare)|(?:tuse|febra|dureri musculare)+$=Ai gripa. Odihneste-te, hidrateaza-te si ia ibuprofen.

^(?=.*durere abdominala)|(?=.*greata)|(?=.*varsaturi)|(?:durere abdominala|greata|varsaturi)+$=>Ai gastroenterita. Ai nevoie de hidratare, repaus alimentar, antiemetice.
^(?=.*dureri de cap)|(?=.*febra)|(?=.*fotofobie)|(?:dureri de cap|febra|fotofobie)+$=>Posibil sa ai meningita. Medicul iti va recomanda antibiotice, analgezice, antipiretice.
^(?=.*durere in gat)|(?=.*tuse)|(?=.*febra)|(?:durere in gat|tuse|febra)+$=>Posibil sa fie amigdalita. Incearca o clatire cu apa sarata, odihna vocala si ia analgezice.
^(?=.*sangerare nazala)|(?=.*dureri de cap)|(?=.*oboseala)|(?:sangerare nazala|dureri de cap|oboseala)+$=>Ai hipertensiune arteriala, Recoman diuretice, beta-blocante, inhibitori ai enzimei de conversie.
^(?=.*dificultati de respiratie)|(?=.*tuse)|(?=.*febra)|(?:dificultati de respiratie|tuse|febra)+$=pneumonie=antibiotice, antiinflamatoare, antipiretice
^(?=.*diaree)|(?=.*greata)|(?=.*varsaturi)|(?:diaree|greata|varsaturi)+$=>Ai o infectie gastro-intestinala, sunt necesare hidratare, repaus alimentar si niste medicamente antiemetice.
^(?=.*greata)|(?=.*varsaturi)|(?=.*durere abdominala)|(?:greata|varsaturi|durere abdominala)+$=>Suspectez o otravire alimentara. E nevoie dehidratare, repaus alimentar, medicamente antiemetice.
^(?=.*durere de urechi)|(?=.*febra)|(?=.*pierdere de auz)|(?:durere de urechi|febra|pierdere de auz)+$=>Probabil ai otita. Medicul iti va recomanda antibiotice, analgezice, picaturi auriculare.
^(?=.*senzatie de arsura la urinare)|(?=.*urinare frecventa)|(?=.*sange in urina)|(?:senzatie de arsura la urinare|urinare frecventa|sange in urina)+$=>Exista posibilitatea sa ai o infectie urinara. Este necesar sa consulti un medic pentru a primi un tratament cu antibiotice.
^(?=.*dureri de cap puternice)|(?=.*greturi)|(?=.*varsaturi)|(?:dureri de cap puternice|greturi|varsaturi)+$=>Este posibil sa suferi de migrena. Poti incerca sa reduci simptomele prin repaus, evitarea factorilor declansatori si medicamente analgezice.